UNIVERSIDAD HISPANOAMERICANA

ESCUELA DE INGENIERÍA INFORMÁTICA

TESINA PARA OPTAR EL GRADO DE BACHILLER EN INGENIERÍA INFORMÁTICA

DESARROLLO E IMPLEMENTACIÓN DE APLICACIÓN WEB PARA LA ONG NAMÁ CONSERVATION EN EL PRIMER SEMESTRE DEL 2021

SUSTENTANTE:

EDWIN EMMANUEL MASÍS SERRANO

TUTOR:

ING. YUSSELIN MURCIA CÉSPEDES

JULIO, 2021

ÍNDICE DE CONTENIDO

Contenido

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

DECLARACIÓN JURADA

DECLARACIÓN JURADA

Yo Edwin Emmanuel Masís Serrano, mayor de edad, portador de la cédula de identidad número 1-1296095 egresado de la carrera de ingeniería informática de la Universidad Hispanoamericana, hago constar por medio de éste acto y debidamente apercibido y entendido de las penas y consecuencias con las que se castiga en el Código Penal el delito de perjurio, ante quienes se constituyen en el Tribunal Examinador de mi trabajo de tesis para optar por el título de bachillerato, juro solemnemente que mi trabajo de investigación titulado: Desarrollo e implementación de aplicación web para la ONG Namú Conservation en el primer semestre del 2021, es una obra original que ha respetado todo lo preceptuado por las Leyes Penales, así como la Ley de Derecho de Autor y Derecho Conexos número 6683 del 14 de octubre de 1982 y sus reformas, publicada en la Gaceta número 226 del 25 de noviembre de 1982; incluyendo el numeral 70 de dicha ley que advierte; articulo 70. Es permitido citar a un autor, transcribiendo los pasajes pertinentes siempre que éstos no sean tantos y seguidos, que puedan considerarse como una producción simulada y sustancial, que redunde en perjuicio del autor de la obra original. Asimismo, quedo advertido que la Universidad se reserva el derecho de protocolizar este documento ante Notario Público.

En fe de lo anterior, firmo en la ciudad de San José, a los 30 días del mes de Junio del año dos mil veintiuno.

Firma del estudiante

Cédula: 1-1296095

CARTA DE APROBACIÓN DEL TUTOR

CARTA DEL TUTOR

Alajuela, 01 de julio de 2021

Sra. María Isabel Losilla
Ingeniería Informática
Universidad Hispanoamericana

Estimada señora:

El estudiante Edwin Emmanuel Masís Serrano cédula de identidad número 1-1296-0995, me ha presentado, para efectos de revisión y aprobación, el trabajo de investigación denominado **"Desarrollo e Implementación de Aplicación Web para la ONG NAMÁ CONSERVATION en el primer semestre del 2021"** el cual ha elaborado para optar por el grado académico de Bachillerato.

En mi calidad de tutor, he verificado que se han hecho las correcciones indicadas durante el proceso de tutoría y he evaluado los aspectos relativos a la elaboración del problema, objetivos, justificación; antecedentes, marco teórico, marco metodológico, tabulación, análisis de datos; conclusiones y recomendaciones.

De los resultados obtenidos por el postulante, se obtiene la siguiente calificación:

a)	ORIGINAL DEL TEMA	10%	8
b)	CUMPLIMIENTO DE ENTREGA DE AVANCES	20%	18
C)	COHERENCIA ENTRE LOS OBJETIVOS, LOS INSTRUMENTOS APLICADOS Y LOS RESULTADOS DE LA INVESTIGACION	30%	28
d)	RELEVANCIA DE LAS CONCLUSIONES Y RECOMENDACIONES	20%	18
e)	CALIDAD, DETALLE DEL MARCO TEORICO	20%	19
	TOTAL		91

En virtud de la calificación obtenida, se avala el traslado al proceso de lectura.

Atentamente,

YUSSELIN TATIANA MURCIA CESPEDES

Firmado digitalmente por YUSSELIN TATIANA MURCIA CESPEDES
Fecha: 2021.07.01 22:20:23 -06'00'

YUSSELIN MURCIA CÉSPEDES
Cédula identidad N 205780828
Carné Colegio Profesional N 9020

CARTA DE APROBACIÓN DEL LECTOR

CARTA DE LECTOR

Universidad Hispanoamericana
Carrera Ingeniería Informática

Dirección

El estudiante Edwin Emmanuel Masis Serrano, cédula de identidad: 1-1296-0995, me ha presentado la Tesina para efectos de revisión y aprobación, denominada "**DESARROLLO E IMPLEMENTACIÓN DE APLICACIÓN WEB PARA LA ONG NAMÁ CONSERVATION EN EL PRIMER SEMESTRE DEL 2021**", el cual ha elaborado para obtener su grado de Bachillerato en Ingeniería Informática.

He revisado y he hecho las observaciones relativas al contenido analizado, particularmente lo relativo a la coherencia entre el marco teórico y análisis de datos, la consistencia de los datos recopilados y la coherencia entre éstos y las conclusiones; así mismo, la aplicabilidad y originalidad de las recomendaciones, en términos de aporte de la investigación. También se realizaron las modificaciones solicitadas a nivel de contenido y forma.

Por consiguiente, este trabajo cuenta con mi aval para ser presentado en la defensa pública.

Atte.

Firma

Nombre Ing. Luis Navarro S

Cédula 2-0484-0537

AUTORIZACIÓN DE PUBLICACIÓN

UNIVERSIDAD HISPANOAMERICANA
CENTRO DE INFORMACION TECNOLOGICO (CENIT)
CARTA DE AUTORIZACIÓN DE LOS AUTORES PARA LA CONSULTA, LA REPRODUCCION PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DE LOS TRABAJOS FINALES DE GRADUACION

San José, 03 de agosto de 2021

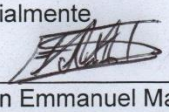
Señores:
Universidad Hispanoamericana
Centro de Información Tecnológico (CENIT)

Estimados Señores:

El suscrito (a) Edwin Emmanuel Masís Serrano con número de identificación 1-1296-0995 autor (a) del trabajo de graduación titulado "Desarrollo e implementación de aplicación web para la ONG Namá Conservation en el primer semestre del 2021" presentado y aprobado en el año 2021 como requisito para optar por el título de bachillerato en ingeniería informática; Si autorizo al Centro de Información Tecnológico (CENIT) para que con fines académicos, muestre a la comunidad universitaria la producción intelectual contenida en este documento.

De conformidad con lo establecido en la Ley sobre Derechos de Autor y Derechos Conexos N° 6683, Asamblea Legislativa de la República de Costa Rica.

Cordialmente,



Edwin Emmanuel Masís Serrano
Cedula. 1-1296-0995

DEDICATORIA

Quiero dedicarle este proyecto a mi madre, quien siempre me ha acompañado durante todos los procesos de mi vida, por haber estado para mí las 24 horas, los 7 días de la semana, por darme apoyo incondicional durante los años de estudio y por enseñarme que hay que ser persistente con la educación porque es la mejor herramienta para cumplir todos las metas que uno se proponga, a mi madre le dedico este proyecto para que lo lleve con orgullo en su corazón, porque gracias a ello voy a llegar a ser un hombre de bien que está preparado para enfrentar los retos de un futuro prometedor.

AGRADECIMIENTO

Deseo agradecer a Dios quien me ha dado la oportunidad de llevar la carrera y por haberme brindado vida y salud para completar este proceso.

A mi madre, quien fue todo un apoyo incondicional, quien ha sido todo un ejemplo de perseverancia, quien me ha enseñado a no rendirme ante las dificultades.

A los profesores de la Universidad Hispanoamericana de Llorente y a mi tutora Yusselin Murcia, quienes aportaron y brindaron el conocimiento para fortalecer mis áreas, logrando retarme a mí mismo para ser mejor cada día.

ABREVIATURAS

ONG: Organización no Gubernamental

MVC: Modelo, vista, controlador

IDE: Entorno de desarrollo integrado

UML: Lenguaje unificado de modelado

API: Interfaz de programación de aplicaciones

SSL: Secure Sockets Layer (Capa de enchufes seguros)

POO: Programación orientada a objetos

SGBD: Sistemas gestores de bases de datos

BD: Bases de datos

HTML: HyperText Markup Language (Lenguaje de marcas de hipertexto)

DHTML: Es la abreviatura de HTML dinámico

MBPS: Megabytes por segundo

CAPÍTULO I: PLANTEAMIENTO DEL TEMA

1.1 ANTECEDENTES Y JUSTIFICACIÓN DEL PROYECTO

1.1.1 Marco de referencia empresarial y contextual

Nombre de la empresa

Namá Conservation

Año de fundación

2019

Misión

Contribuir activamente a la conservación de carnívoros, sus presas y ecosistemas en Costa Rica, mediante el desarrollo de investigación y colaboración con aliados nacionales e internacionales, aportando valiosa información para la toma de decisiones de conservación a nivel local y nacional.

Visión

Ser una organización líder en investigación y conservación de carnívoros, sus presas y ecosistemas en Costa Rica, reconocida por ser una organización interdisciplinaria e inclusiva con las comunidades y los diferentes actores en los procesos de conservación de la diversidad biológica, fortaleciendo el capital humano con un enfoque de lo local a lo global.

Estrategia

Nos enfocamos en las personas y comunidades locales para empoderarlas en conservación con actividades económicas y productivas amigables con el ambiente.

Objetivos

- Generar iniciativas para conservar los ecosistemas de importancia para los carnívoros silvestres en el país con enfoques que incluyan el beneficio de la calidad de vida de los seres humanos.
- Generar información de relevancia biológica, ecológica, ética, histórica y ambiental a nivel local y nacional, para ayudar en la toma de decisiones en pro de la conservación de la biodiversidad.
- Promover la participación de las comunidades locales en la tarea de conservación de los hábitats donde habitan los felinos y sus presas.
- Propiciar la generación y divulgación de información sencilla para crear conocimiento sobre las especies, ecosistemas y relación con los seres humanos.
- Contribuir a la mejora de la sensibilización sobre la importancia del rol ecológico que cumple la vida silvestre en la cotidianeidad del ser humano.
- Realizar alianzas nacionales e internacionales para el desarrollo de investigaciones de alto nivel que se traduzcan en la conservación de nuestras especies focales.
- Desarrollar campañas de voluntariados en temas que favorezcan la conservación de los ecosistemas de la vida silvestre.

Organización

Namá Conservation es una organización costarricense sin fines de lucro enfocada en la conservación de los ecosistemas naturales del país, sus especies clave y la integridad de las comunidades que viven en interacción con ellas.

Desarrollamos investigación ecológica para proporcionar una línea de base que sirva para la toma de acciones de conservación de especies y ecosistemas en conjunto con proyectos de desarrollo humano.

Historia de la organización

Namá Conservation surge como una iniciativa de un grupo de científicos conservacionistas y personas de la sociedad civil interesadas en contribuir a la conservación de carnívoros en el país. Esta iniciativa pretende desarrollar proyectos de investigación y conservación con un impacto real a lo largo del país partiendo de las necesidades locales a las globales.

Gracias a los 30 años de trabajo y experiencia de nuestros científicos en planta, hemos podido combinar la experiencia con la pasión de nuevas generaciones conservacionistas para poder interactuar con comunidades y pueblos originarios. Es en estos territorios donde las especies y ecosistemas más amenazados se ubican, al mismo tiempo que coexisten pobladores locales a lo largo de grandes paisajes enfrentando múltiples presiones antropogénicas. Estos sitios por excelencia son bioculturales, donde se integra la naturaleza y la cultura. Por lo que en honor a uno de los pueblos originarios más grandes de Costa Rica representamos nuestra filosofía “de lo local a lo global”, plasmando nuestro nombre como una asociación consolidada que parte de la cultura autóctona Cabécar. Donde la palabra Namá representa a las seis especies de felinos presentes en Costa Rica.

1.1.2 Justificación del proyecto

La ONG Namá Conservation requiere de una aplicación web para poder recibir con prioridad las donaciones que realizan las diferentes personas alrededor del mundo, es decir, eficientizar el proceso actual de recaudación de fondos dados por los internautas, con el fin de continuar desarrollando proyectos y que estos trabajos realizados por los voluntariados sean mostrados al público, para continuar impulsando las donaciones y actividades que implican una trasparencia absoluta en la gestión de cualquier fundación.

Con la aplicación web funcionando, Namá Conservation podrá convertir en realidad, la premisa de que la totalidad de las aportaciones monetarias que ingresan a la ONG van destinadas para el funcionamiento del conservatorio, logrando optimizar el proceso actual de recaudación de donaciones y mejorando el posicionamiento en la web, donde podrán comunicar de forma eficaz

los proyectos para que los internautas puedan obtener una mayor credibilidad a la hora de realizar alguna donación monetaria.

1.2 DEFINICIÓN DEL PROBLEMA

1.2.1 Problemática

Actualmente, la ONG Namá Conservation tiene problemas en el proceso de poder recibir los donativos monetarios en forma eficaz que le realizan los navegantes de internet, esto debido a que no cuentan con un sistema centralizado o una página web donde exista la posibilidad de que cualquier usuario pueda realizar los aportes caritativos de forma eficiente.

Namá Conservation, utiliza las redes sociales como Facebook o Instagram, para mostrar sus labores, donde su principal interés es que los internautas puedan apreciar los trabajos o proyectos realizados por el conservatorio y que estos motiven al cliente a realizar una donación para la continuidad de la fundación.

Una vez que el cibernauta decide realizar una caridad, las redes sociales redirigen al consumidor a un cuestionario en Google forms, donde el usuario ingresa los datos con el fin de ser contactado posteriormente por el conservatorio para poder realizar la donación.

Esta metodología actualmente está dañando la recaudación de fondos del conservatorio, ya que no siempre se logra contactar con el donador por diferentes temas como la diferencia horaria, datos erróneos u otros procesos engorrosos.

Ante lo anterior, la principal consecuencia de estos problemas es que se generan pérdidas monetarias causando afectación en la financiación y la sostenibilidad de los proyectos de la organización.

Diagrama causa y efecto

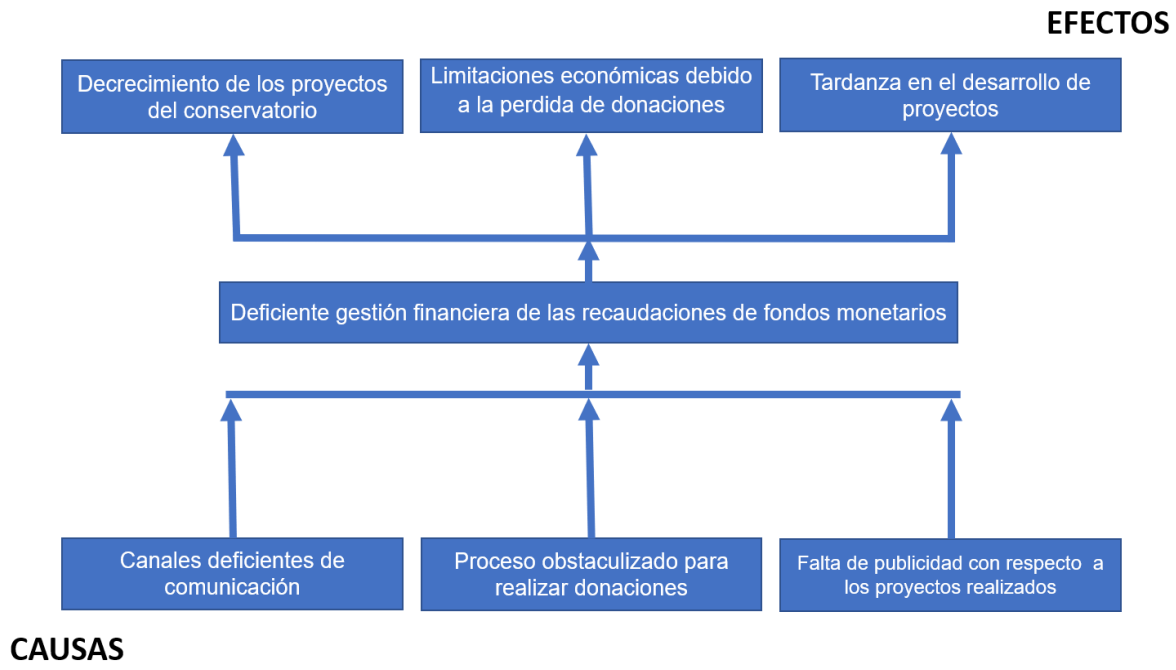


Figura 1 Diagrama de causa-efecto
Fuente: Elaboración propia

1.2.2 Problema general

¿Cómo se puede eficientizar el proceso de recaudación de las donaciones monetarias para poder efectuar una mejora financiera en la gestión de proyectos de Namá Conservation en el primer semestre del 2021?

1.2.3 Problemas específicos

¿Cuál es el proceso actual de Namá Conservation para poder recibir las donaciones monetarias y las solicitudes de participación de voluntariados?

¿Cuáles son los requerimientos que se deben identificar en los procesos de voluntariados y donaciones para el desarrollo de la aplicación web para Namá Conservation?

¿Cuál es la mejor forma de optimizar los procesos de recepción de voluntariados y de donaciones de Namá Conservation?

¿Cuál es la solución para mejorar el proceso de recepción de donaciones y solicitudes de voluntariados del conservatorio?

¿Cómo Namá Conservation puede obtener mayor credibilidad con los cibernautas que pueden ser donadores potenciales?

1.3 OBJETIVOS DEL PROYECTO

1.3.1 Objetivo general

Desarrollar una herramienta web para la gestión de caridades monetarias que permita contribuir y mejorar el proceso de la recaudación de las donaciones para impulsar la sostenibilidad del desarrollo de los proyectos del conservatorio.

1.3.2 Objetivos específicos

- Analizar e identificar el proceso actual de la organización con respecto a la gestión para recibir donaciones y solicitudes de voluntariados, mediante técnicas o herramientas de recolección de datos.
- Identificar los requerimientos funcionales y no funcionales junto con las necesidades que se deben contemplar en el proyecto.
- Diseñar los diagramas de base de datos y los diagramas de casos de uso, del nuevo proceso de recolección de donativos y solicitudes de voluntariados que permita cubrir las necesidades plasmadas en los requerimientos mediante lenguaje UML.
- Desarrollar el sistema web para mejorar el proceso de recaudación de donaciones, publicaciones y solicitudes de voluntariados bajo el patrón de arquitectura MVC.
- Implementar la aplicación web desarrollada en el hosting, con el fin de poner en marcha la solución del problema principal.

1.4 ALCANCES Y LIMITACIONES

1.4.1 Alcances

A continuación, se detallan los entregables del proyecto:

- El primer entregable del proyecto es el diagnostico actual de Namá Conservation basado en la información captada mediante las herramientas de recolección de datos.
- El segundo entregable del proyecto es el listado de los requerimientos funcionales y no funcionales que el sistema debe tener para cumplir con las funcionalidades de la aplicación web.
- El tercer entregable del proyecto son los diagramas de base de datos y diagramas de casos de uso mediante el lenguaje UML, donde se encuentre plasmado el nuevo procedimiento para realizar los procesos de donaciones y solicitudes de voluntariado.
- El cuarto entregable del proyecto es la presentación y la descripción del funcionamiento de los módulos de donaciones, voluntariados y publicaciones.
- El ultimo entregable del proyecto es la publicación de la aplicación web.

1.4.2 Limitaciones

El proyecto cuenta con las siguientes limitaciones:

- El proyecto se desarrollará con herramientas de código abierto, por lo tanto, se deberá realizar con el IDE Visual Studio Code que es un software libre, bajo la licencia MIT (Una licencia permisiva breve y simple con condiciones que solo requieren la preservación de los derechos de autor y avisos de licencia)
- La base de datos se realizará con MySQL que es una base de datos de código abierto, escalable y compatible con los navegadores web.
- El proyecto debe tener integración con PayPal para poder realizar transacciones monetarias, esta integración tiene una cobertura en 190 países del mundo (Existen algunos países donde no está habilitado este servicio) Además, Namá Conservation se hará cargo del coste de las transacciones.

- Namá Conservation cuenta y asume los costos con respecto al servicio de hosting, dominios y certificados SSL.
- El proyecto se desarrollará bajo la arquitectura MVC con el framework de Laravel, debido a que Namá Conservation tiene un socio comercial, quien brindará actualizaciones y soporte una vez finalizado el proyecto.

1.5 CRONOGRAMA DEL PROYECTO

Nombre de tarea	Duración	Comienzo	Fin
Proyecto Namá Conservation	**97 días**	**lun 1/18/21**	**mar 6/1/21**
Documentación	**6 días**	**lun 1/18/21**	**lun 1/25/21**
Definicion de requerimientos	3 días	lun 1/18/21	mié 1/20/21
Documentación del proyecto	3 días	jue 1/21/21	lun 1/25/21
Base de datos	**3 días**	**mar 1/26/21**	**jue 1/28/21**
Estructura y desarrollo de la base de datos	3 días	mar 1/26/21	jue 1/28/21
Módulo de Voluntariado	**21 días**	**vie 1/29/21**	**vie 2/26/21**
Desarrollo de la estructura interna del proyecto	3 días	vie 1/29/21	mar 2/2/21
Desarrollo del backend del módulo de voluntariado	7 días	mié 2/3/21	jue 2/11/21
Desarrollo de las vistas del módulo de voluntariado	7 días	vie 2/12/21	lun 2/22/21
Conexión del Backend y frontend del módulo de voluntariado	4 días	mar 2/23/21	vie 2/26/21
Módulo de Publicaciones	**17 días**	**lun 3/1/21**	**mar 3/23/21**
Desarrollo del backend del módulo de publiciones	7 días	lun 3/1/21	mar 3/9/21
Desarrollo de las vistas del módulo de publicaciones	7 días	mié 3/10/21	jue 3/18/21
Conexión del Backend y frontend del módulo de publicaciones	3 días	vie 3/19/21	mar 3/23/21
Creación de página web	**10 días**	**mié 3/24/21**	**mar 4/6/21**
Desarrollo de las páginas estaticas de la empresa	10 días	mié 3/24/21	mar 4/6/21
Módulo de Donaciones	**28 días**	**mié 4/7/21**	**vie 5/14/21**
Desarrollo del backend del módulo de donaciones	15 días	mié 4/7/21	mar 4/27/21
Desarrollo de las vistas del módulo de donaciones	7 días	mié 4/28/21	jue 5/6/21
Conexión del Backend y frontend del módulo de donaciones	6 días	vie 5/7/21	vie 5/14/21
Cierre de proyecto	**12 días**	**lun 5/17/21**	**mar 6/1/21**
Pruebas finales	7 días	lun 5/17/21	mar 5/25/21
Creación del manual de usuario	4 días	mié 5/26/21	lun 5/31/21
Presentación del proyecto	1 día	mar 6/1/21	mar 6/1/21

Tabla 1 Cronograma del proyecto
Fuente: Elaboración propia

CAPÍTULO II: MARCO TEÓRICO

2.1 INTRODUCCIÓN

El presente proyecto está enfocado en el desarrollo de una aplicación web que permita facilitar el proceso de las recaudaciones de donaciones monetarias mediante un software para una ONG. Para ello, es importante desarrollar en este capítulo, todas las definiciones o conceptos con los que se desarrolló el proyecto, empezando con el concepto de software, explicando su ciclo de desarrollo y herramientas junto con la metodología aplicada para lograr como resultado final, la publicación de la aplicación web dentro del internet.

A continuación, se detallan cada uno de esos conceptos:

2.2 SOFTWARE

> El software es el componente lógico y la parte intangible del ordenador que permite interaccionar con el hardware. A pesar de ser intangible es tan importante como el hardware, estando estrechamente relacionados, tanto, que el uno sin el otro no podría funcionar. (Aranda, 2016, p.12)

Como lo explica el autor Aranda, el software es una parte fundamental de la computadora ya que, sin él, el equipo (sea computadora o dispositivo móvil) no podría funcionar de forma adecuada. El software está presente en muchos aspectos de la tecnología y cada uno cumple con una función específica, desde ser un sistema operativo para el levantamiento de las operaciones de una computadora, o el funcionamiento de los programas de Excel o Word que se usan regularmente en una computadora.

2.3 INGENIERÍA DEL SOFTWARE

De acuerdo con Sommerville (2011) “La ingeniería del software es una disciplina de ingeniería que se interesa por todos los aspectos de la producción de software, desde las primeras etapas de la especificación del sistema hasta el mantenimiento del sistema después de la operación.” (p.25), en otras palabras, los ingenieros de software se dedican a realizar los programas (en algunos casos desde cero), levantando las necesidades de los clientes, plasmadas en

requerimientos, donde se preocupan por desarrollar una arquitectura para que el software pueda funcionar de forma eficiente. Además, se aplican una serie de técnicas y prácticas para facilitar la solución a los problemas que puedan aparecer durante la etapa del desarrollo.

2.4 LENGUAJE DE PROGRAMACIÓN

Los lenguajes de programación vienen siendo como el idioma en el que fue creado cada programa. Normalmente los programas instalados en los dispositivos están realizados en diferentes idiomas según las necesidades, ya que este es el medio de comunicación con el lenguaje de máquina para que el dispositivo pueda interpretar las órdenes y así poder ejecutar el software.

Según el autor Casado: "Un lenguaje de programación es un conjunto de instrucciones, operadores y reglas de sintaxis y semánticas, que se ponen a disposición del programador para que éste pueda comunicarse con los dispositivos de hardware y software existentes" (Casado, 2015, p.13)

2.4.1 Visual Studio Code

Visual Studio Code es un editor de código fuente que permite crear, modificar y hasta ejecutar las aplicaciones web con extensiones para PHP, JavaScript, Node.js entre otros. Este se ejecuta desde el escritorio de la computadora y está disponible para varios sistemas operativos.

2.5 PROGRAMACIÓN ORIENTADA A OBJETOS

La programación orientada a objetos es una forma de captar los diferentes elementos del diario vivir del software a crear, donde se trata de simular los objetos, personas, animales de la realidad para así poder plasmar las necesidades o procesos que el software debe de realizar. Para lograr esa simulación es importante tomar en cuenta los siguientes conceptos:

2.5.1 Objetos y clases

> Una clase en Programación Orientada a Objetos actúa como una plantilla, modelo, o prototipo, a partir de la cual se obtienen instancias, habitualmente llamadas objetos. Dichos objetos son como "clones" o copias repetidas que ocuparán un espacio de memoria diferente cada uno de ellos. (Blasco, 2019, p.89)

2.5.2 Abstracción

Según el autor Oviedo define abstracción como "La acción de identificar las cualidades y acciones que un objeto puede realizar, lo que permite diferenciar los conceptos de clase y objeto" (Oviedo, 2015, p.222)

2.5.3 Herencia

El autor Oviedo, define la herencia con la siguiente analogía:

> Al igual que los hijos heredamos características de nuestros padres, en la POO podemos obtener clases hijas con el mismo contenido de su clase padre. De esta manera se obtiene una reutilización de las clases, la cual es una de las propiedades más deseadas en la programación ya que conlleva un ahorro de tiempo y de líneas de código. (Oviedo, 2015, p. 257)

La aplicación de la herencia en la programación orientada a objetos permite generar una jerarquía de clases, donde las mismas permiten llamar a cualquier objeto o clase que este dentro de la herencia, para que de esta forma el software logre obtener la información de dicha clase de una forma eficiente.

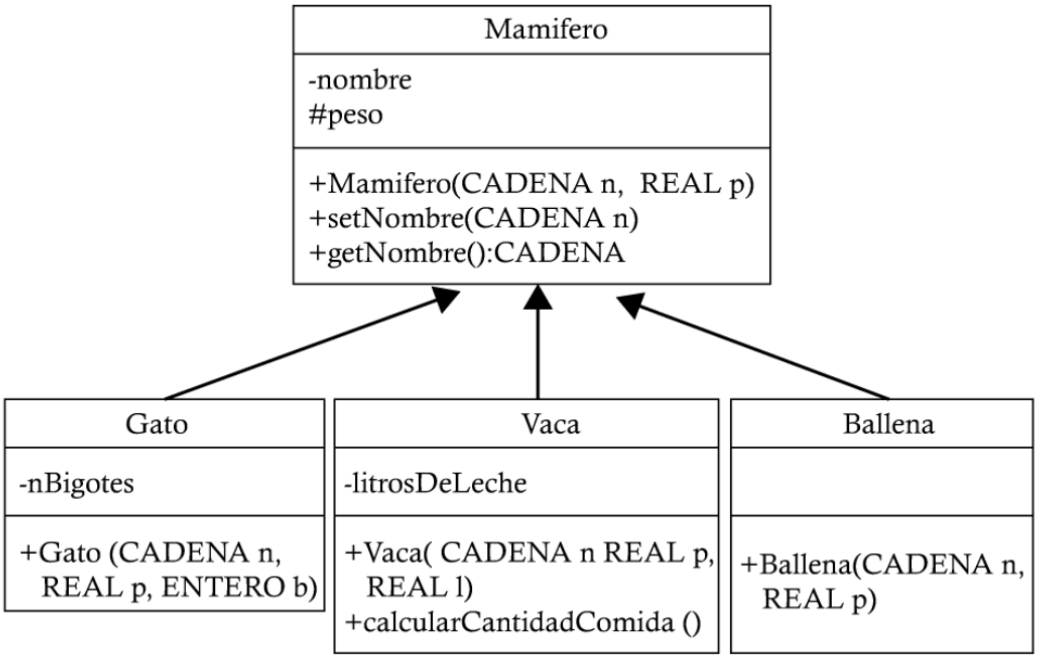


Figura 2 Diagrama de clases
Fuente: Lógica de programación orientada a objetos

2.5.4 Encapsulamiento

El autor Oviedo, define el encapsulamiento como “la propiedad que tienen las clases de agrupar las características y las acciones relacionadas con una abstracción bajo una misma unidad de programación” (Oviedo, 2015, p.222)

2.5.5 Polimorfismo

El autor Blasco explica el polimorfismo de la siguiente forma:

> El propio término, “polimorfismo” implica múltiples formas. En una aplicación polimórfica nos encontramos con una “familia” de clases, todas ellas con un “ascendiente” común, que marca un patrón de comportamiento genérico, igual para todas las clases “descendientes”, pero que, partiendo de ese patrón de comportamiento común, cada una de esas clases descendientes (SubClases) le puede dar una “forma” diferente, es

decir, una implementación específica concreta. De ahí lo de muchas formas, es decir, poliforma. (Blasco, 2019, p.329).

2.6 PROGRAMACIÓN WEB

2.6.1 Servidores web

Los servidores web son un grupo de computadoras donde su principal función es el almacenamiento de las páginas web, junto con la recepción y envió de peticiones por parte del cliente (Cibernauta), los autores Vara, Verde y López, los definen de la siguiente forma:

> Su cometido principal es proveer de contenido estático a un cliente que ha realizado una petición a través de un navegador. Para ello, carga un archivo y lo sirve a través de la red al navegador del solicitante. Este es el funcionamiento habitual cuando lo que se implementa en el servidor es una aplicación web estática (HTML) o dinámica (DHTML), es decir, que el servidor se convierte en un instrumento que proporciona un lugar para guardar y administrar los recursos HTML, que pueden ser accesibles por los usuarios de la red a través de navegadores. (Vara Mesa, Verde Marín, López Sáenz, 2015, p.20)

Estos servidores conectados a internet logran establecer la comunicación mediante los protocolos de conexión HTTP, TCP, IP entre otros.

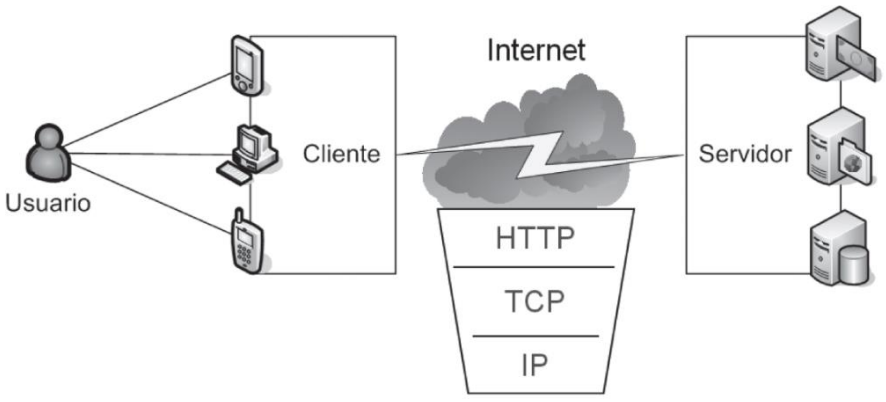


Figura 3 Diagrama del modelo cliente/servidor
Fuente: Desarrollo web en entorno servidor

2.6.2 Protocolo TCP/IP

Según el autor Carvajal:

> El protocolo TCP/IP es un conjunto de reglas y normas que nos sirven para comunicar dos o más ordenadores entre sí; en esas reglas se especifica como se van a interpretar los datos que se reciban y la forma de enviarlos. El protocolo TCP/IP es un protocolo fiable, seguro, rápido y práctico; es adecuado para todo tipo de redes, aunque está orientado a redes locales y extensas. (Carvajal, 2017, p.25)

2.6.3 Protocolo HTTP

El protocolo HTTP actualmente es el más utilizado para la navegación web, de acuerdo con Carvajal (2017) “este se caracteriza por permitir conexiones persistentes, además de las posibilidades, por parte de cliente, de enviar varias peticiones en una misma conexión” (p.44) donde el mismo funciona dentro de la capa de aplicación del modelo TCP/IP, según Carvajal (2017) “Cuando el cliente realiza una petición a una página web, el servidor encuentra el objeto deseado y lo envía (además de otra información útil como el tipo de objeto, longitud, etc.) al navegador como respuesta HTTP” (p.46) estas respuesta es a lo comúnmente se llama como request o response de los diferentes tipos de peticiones.

2.6.4 Tipos de peticiones

Las peticiones son métodos para poder realizar la comunicación entre la computadora del cliente o cibernauta con el servidor donde se aloja la página web. Las 4 peticiones más comunes son las siguientes:

- Método GET: Este método es utilizado para poder solicitar al servidor información con respecto al almacenamiento y visualización de una página web, solicitar una respuesta con información entre otras cosas.

- Método POST: Este método es utilizado para el envío de información desde el cliente hacia el servidor, principalmente es utilizado para el envío de formularios web, login entre otras solicitudes.
- Método PUT: Este método es utilizado para la actualización de algún dato que esté presente en el servidor.
- Método DELETE: El método DELETE es utilizado para poder eliminar información que esté presente en el servidor. Para efectos del proyecto, la información no es eliminada del servidor, sino que es ocultada para el cliente.

2.6.5 HMTL

Según el autor Terán (2016) "un documento HTML es un archivo ASCII o archivo de texto plano, que mediante una serie de marcas o etiquetas va definiendo los elementos que lo componen. Para crear un archivo ASCII se puede utilizar cualquier editor de texto. (p.28) en otras palabras; es un lenguaje de etiquetas que es utilizado para la creación de las páginas web.

2.6.6 JavaScript

JavaScript es un lenguaje que permite darle funcionalidad a las páginas de HTML, o como lo describe el autor Recio:

> JavaScript es un lenguaje ejecutado por el navegador. Su sintaxis es similar a la del archiconocido lenguaje de programación Java –de ahí su nombre–. La idea básica de JavaScript es permitir definir acciones cuando ocurren ciertos eventos en el navegador: la página se ha cargado completamente, el usuario hace un clic, etc. (Recio, 2016, p.25)

2.6.7 CSS

Las CSS (cascade style sheets) (Recio, 2016) "sirven para indicar el formato de cualquier elemento de nuestro código HTML. Simplemente obviamos toda la información sobre el formato

en el código HTML e indicamos el fichero CSS donde la hemos centralizado" (p.23) en otras palabras las hojas CSS ayudan a darle diseño, colores, formas o hasta animaciones al contenido del HTML.

2.6.8 PHP

Según los autores Vara y Verde:

> PHP (HyperText Processor) es uno de los lenguajes más extendidos actualmente. Cuenta con una comunidad de desarrolladores muy importante debido a sus características de gratuidad, código abierto, la posibilidad de ser portado y ejecutado en diferentes plataformas, etc. se define como un lenguaje imperativo de tipos dinámicos, con la posibilidad de utilizar construcciones orientadas a objetos. es soportado por la gran mayoría de servidores web actuales." (Vara y Verde, 2015, p.18)

2.6.9 API

En la actualidad existen muchas integraciones de funciones para las páginas web por medio de API, según Eslava "Una Interfaz de Programación de Aplicaciones (o API de sus siglas en inglés), define las clases, métodos, funciones y variables que la aplicación necesita llamar para realizar una tarea" (2018, p.10) por medio de las API los desarrolladores logran conectar diferentes servicios dinámicos para la integración de una página web que carezca de tal funcionalidad.

2.7 BASES DE DATOS

La base de datos es un sistema de gestión de información que permite almacenar toda la información del software, o como mencionan los autores Pulido y Escobar (2019) "Una base de datos (cuya abreviatura es BD) es una colección de información organizada de tal modo que sea fácilmente accesible, gestionada y actualizada" (p.29) donde sus principales características son:

- Independencia de los datos. Significa que los datos no dependen del programa y, por tanto, cualquier aplicación puede hacer uso de ellos.
- Reducción de la redundancia. Llamamos redundancia a la duplicidad de los datos. Cuando ésta se reduce al máximo, conseguimos un mayor aprovechamiento del espacio y además evitamos que existan inconsistencias entre los datos.
- Seguridad. Es la protección de la base de datos frente a usuarios no autorizados. (Pulido y Escobar, 2019, p.29)

2.7.1 MYSQL

MYSQL es un sistema de gestión de base de datos que es la más "elegida por la gran mayoría de programadores en PHP. Soporta el lenguaje SQL y la conexión de varios usuarios" (Pavón, 2014, p.17)

> Además, el motor de datos de MySQL es mucho más rápido, tanto grabando datos como localizándolos y recuperándolos, que el de otras bases de datos. Eso sin contar con que MySQL ofrece una gran seguridad sobre la integridad de los datos almacenados. (López, 2014, p.33)

Este SGBD es muy funcional para el uso framework de Laravel, debido a que utiliza un sistema de migraciones de base de datos, que en otras palabras es, un versionado de los cambios de la BD permitiendo que la aplicación sea modificable y escalable a futuro.

2.7.2 MONGO DB

> Mongo DB es una base de datos no relacional (Que no tabula la información por filas y columnas) esta trabaja por medio de documentos que son "valores, y que se corresponden con algunas estructuras de datos típicas de los lenguajes de programación tales como tablas hash o diccionarios. En general, los documentos contendrán múltiples pares clave-valor, como, por ejemplo, {"Nombre":"Juan","País":"España"}.". (Sarasa, 2016, p.47)

2.7.3 POSTGRESQL

PostgreSQL es un:

> Sistema de base de datos posrelacional de código abierto gratuito que se ejecuta en todos los sistemas operativos principales. La historia del desarrollo de PostgreSQL durante 25 años proporciona una enorme gama de funciones para desarrolladores y administradores de bases de datos (DBA), entregadas en un robusto servidor de software utilizado en todo el mundo. (Capterra, s.f.)

2.7.4 MICROSOFT SQL SERVER

> "Es un sistema de gestión de base de datos relacional desarrollado como un servidor que da servicio a otras aplicaciones de software que pueden funcionar ya sea en el mismo ordenador o en otro ordenador a través de una red" (Parada, 2019)

2.8 FRAMEWORK

Según el autor Pérez, define framework como:

> En inglés framework se puede traducir como estructura. En el sentido que nos ocupa un framework sería un marco de trabajo. MVC son las siglas del Modelo-Vista-Controlador, un paradigma de programación de aplicaciones que separa en tres niveles el trabajo:
>
> - El modelo: Especifica la forma de manipular los datos por parte de la aplicación. Es decir, especifica cómo son los datos (qué tipo tienen) y la forma de manipularlos. Este modelado de datos enlaza con la lógica de negocio, es decir, con la forma en la que los datos se almacenan en la capa de negocio (en la base de datos, en definitiva).
> - La vista: Hace referencia al aspecto visual de la aplicación de cara al usuario, especifica la forma de interaccionar que tendrá la aplicación con el usuario.

- El controlador. Es la parte que controla las acciones del usuario y las comunica a los dos niveles anteriores. (2015, p.83)

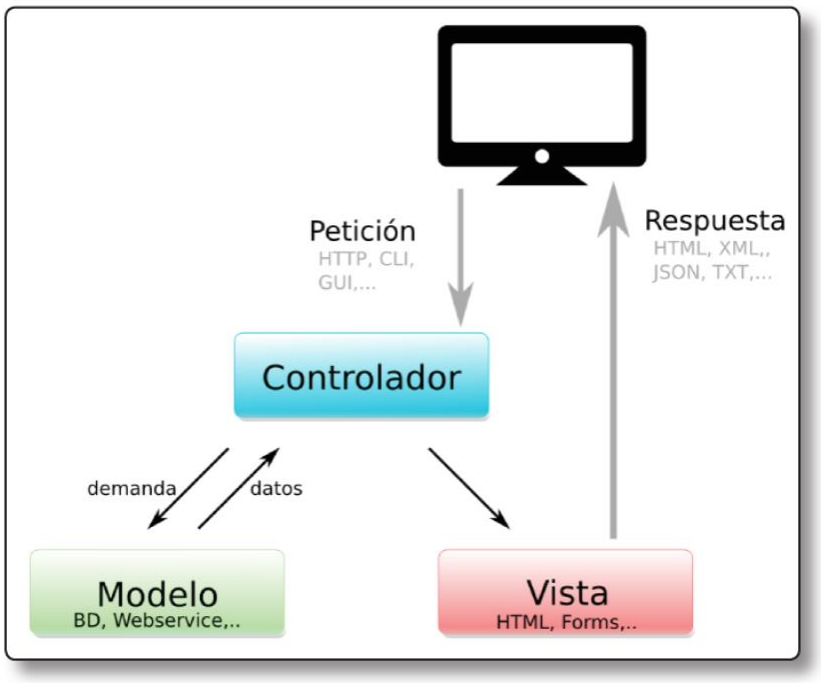


Figura 4 Diagrama de MVC
Fuente: Desarrollo de aplicaciones mediante el Framework de Spring p.84

2.8.1 Laravel

Laravel es un framework de código abierto enfocado en el desarrollo de aplicaciones web, especialmente PHP con una extensa comunidad, que ofrece las funcionalidades necesarias para desarrollar aplicaciones actuales. Laravel contiene un enfoque de desarrollo sencillo y de forma elegante, ya que contiene sus propios métodos para realizar consultas o conexiones de base de datos, integraciones con múltiples librerías y un ecosistema muy estable para el desarrollo. Este framework nació con influencias de frameworks como Ruby on Rails o ASP.NET MVC y fue creado por Taylor Otwell en el 2011.

Hoy en día es el framework más utilizado para PHP, se destaca por su interfaz elegante, sencilla y fácil de usar, o como lo menciona Dorantes (2015) “No importa si eres un experto en PHP o si son tus primeros pasos; cuando lo conozcas, sabrás que Laravel es el framework que estabas

buscando para tus proyectos PHP" (párr.4) esto debido a que Laravel contiene servicios de autenticación de usuarios, validación en formularios, accesos a la base de datos con sus propios métodos y una serie de plantillas realizadas en diferentes librerías que ayudan a simplificar el trabajo de los programadores.

Otras de las características más importantes de Laravel son:

- HTTP Middleware de Laravel proporcionan un mecanismo conveniente para filtrar las peticiones HTTP entrantes a una aplicación. Laravel incluye un middleware que permite verificar si un usuario está autentificado cuando acceda a la aplicación. Si el usuario no lo estuviera, el middleware lo redireccionaría a la pantalla de login. Y, por el contrario, si lo estuviera, el middleware permitiría el acceso a la aplicación. (Rondón, 2016, párr.1)
- HTTP Routes es "un servicio de Laravel que permite construir nuestras rutas. Los métodos del servicio se relacionan con los métodos de petición del protocolo HTTP, también conocidos como verbos HTTP" (Cíceri, 2018, p.64)
- Object-Relational-mapping (mapeo de objeto-relacional) es un modelo de programación que consiste en la transformación de las tablas de una base de datos, en una serie de entidades que simplifiquen las tareas básicas de acceso a los datos para el programador.
- Tareas automatizadas, Laravel puede crear tareas para que cada cierto tiempo sean ejecutadas por el servidor.
- Las colas de Laravel proporcionan una API unificada en una variedad de diferentes backends de cola, como Beanstalk, Amazon SQS, Redis o incluso una base de datos relacional. Las colas le permiten aplazar el procesamiento de una tarea que consume tiempo, como enviar un correo electrónico, hasta un momento posterior. Aplazar estas tareas que consumen mucho tiempo acelera drásticamente las solicitudes web a su aplicación (Laravel, s.f.)

2.8.2 Asp.NET

Según el autor Ceballos, Visual Studio .NET es:

Un entorno de desarrollo multilenguaje diseñado por Microsoft para simplificar la construcción, distribución y ejecución de aplicaciones para Internet. Tiene fundamentalmente tres componentes: una máquina virtual (CLR: Common Language Runtime) que procesa código escrito en un lenguaje intermedio (MSIL: Microsoft Intermediate Language), una biblioteca de clases (biblioteca .NET) y ASP.NET que proporciona los servicios necesarios para crear aplicaciones Web. (Ceballos, 2015, p.24)

2.8.3 Ruby on Rails

Ruby on Rails es un marco de trabajo cimentado en el patrón Modelo-Vista-Controlador o MVC. Este patrón permite la organización de las partes del software desarrollado en función de la finalidad de cada una. El objetivo de estas divisiones es claro: evitar que cuando se realicen cambios en una parte del código estos perjudiquen a otra parte. (Maldonado, 2018, párr.4)

2.8.4 CodeIgniter

CodeIgniter es un framework PHP que usa una arquitectura de Model View Controller (MVC). En términos sencillos, eso significa que CodeIgniter utiliza diferentes componentes para manejar tareas de desarrollo específicas. Este enfoque es muy popular entre los desarrolladores porque permite crear aplicaciones web altamente escalables con un tamaño más reducido. (B, 2020)

2.8.5 Symfony

Symfony es un entorno de trabajo estandarizado (framework PHP) que se utiliza para el desarrollo de aplicaciones web y es de los más utilizados en el entorno de desarrolladores de apps. En otras palabras, es una herramienta para desarrolladores para crear aplicaciones en PHP. (B, 2020)

2.9 DESARROLLO DEL SOFTWARE

2.9.1 Ciclo de programación

El desarrollo de un software o de un conjunto de aplicaciones pasa por diferentes etapas desde que se produce la necesidad de crear un software hasta que se finaliza y está listo para ser usado por un usuario. Ese conjunto de etapas en el desarrollo del software responde al concepto de ciclo de vida del programa. No en todos los programas ni en todas las ocasiones el proceso de desarrollo llevará fielmente las mismas etapas en el proceso de desarrollo; no obstante, son unas directrices muy recomendadas. (Casado, 2015, p.18)

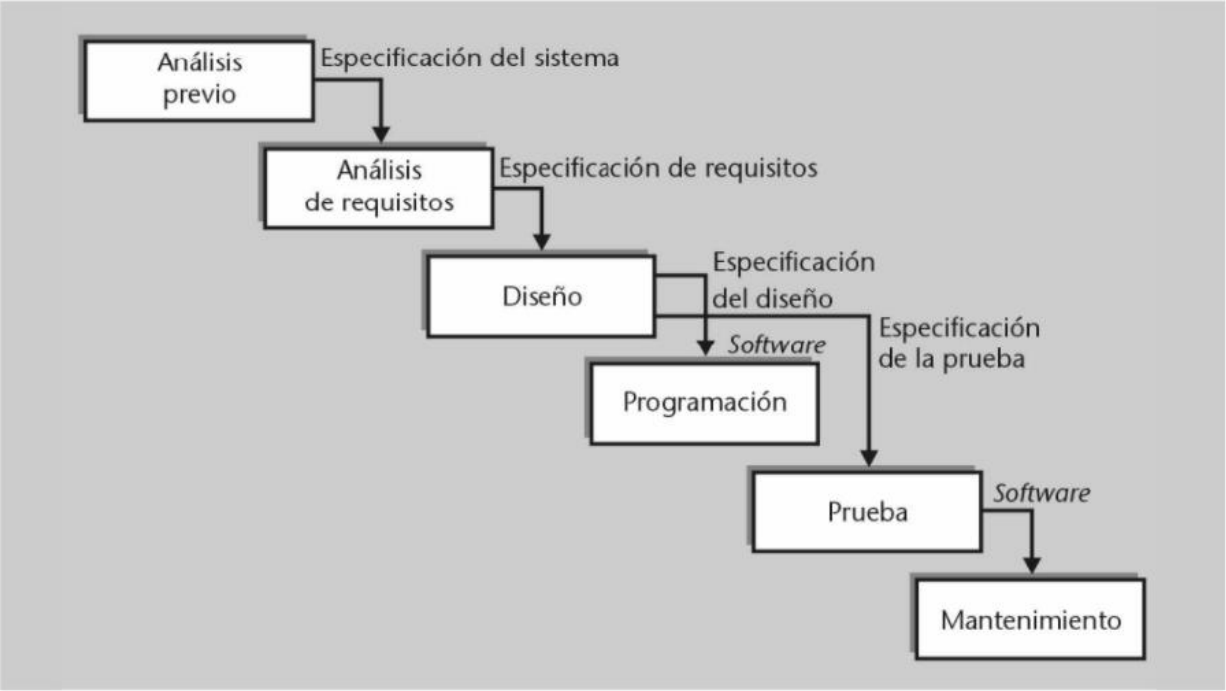


Figura 5 Diagrama del ciclo de la programación
Fuente: ingeniería del software p.21

2.9.1.1 Análisis

> La fase de análisis define los requisitos del software que hay que desarrollar. Inicialmente, esta etapa comienza con una entrevista al cliente, que establecerá lo que quiere o lo que cree que necesita, lo cual nos dará una buena idea global de lo que necesita, pero no necesariamente del todo acertada. Aunque el cliente crea que sabe lo que el software tiene que hacer, es necesaria una buena habilidad y experiencia para reconocer requisitos incompletos, ambiguos, contradictorios o incluso necesarios**.** (Casado, 2015, p.19)

En esta etapa de análisis es importante detectar cual es la verdadera necesidad que tiene el cliente, para ello, se utilizan una serie de herramientas que ayudan comprender la funcionalidad que el software debe de tener. Entre ellas están:

Las Historias de usuario

Las historias de usuario funcionan para "describir las funcionalidades que va a tener la aplicación, y la manera en que los usuarios interactuarán con las mismas. Para definir claramente las historias de usuario, se deben especificar los requerimientos priorizados y refinados" (Molina, 2019, p.77)

Desarrollo ágil: Historias de usuario y criterios de aceptación

	Enunciado de la Historia				Criterios de Aceptación		
Identificador (ID) de la Historia	Rol	Característica / Funcionalidad	Razón / Resultado	Número (#) de Escenario	Criterio de Aceptación (Título)	Contexto	Evento
XX-XXXX-XXXX	Como un [Rol]	Necesito [Descripción de la Funcionalidad]	Con la finalidad de [Descripción razón o resultado]	1	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]
				2	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]
				3	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]
				4	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]
XX-XXXX-XXXX	Como un [Rol]	Necesito [Descripción de la Funcionalidad]	Con la finalidad de [Descripción razón o resultado]	1	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]
				2	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]
				3	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]
				4	[Titulo del escenario]	En caso que [Contexto] y/ o [Contexto]	cuando [Evento]

Tabla 2 Ejemplo de historia de usuario
Fuente: La oficina de proyectos en informática

Casos de uso

Los casos de uso describen “las interacciones entre los actores y el sistema que se produce durante la ejecución de ciertas características que capturan una funcionalidad ejecutable de la línea de productos, y seria por tanto el lugar adecuado para plasmar la variabilidad anterior” (Garzás y Piattini, 2015, p.239)

[Nombre de Caso de Uso Nro. 1]

Caso de Uso	[Nombre del Caso de Uso]	**Identificador:** [Del caso de uso]
Actores	[Listado de los actores que tienen participación en el caso de uso]	
Tipo	[Tipo de caso de uso, primario, secundario, opcional]	
Referencias	[Requerimientos o funcionalidades incluidas en este caso de uso. Casos de uso relacionados.]	
Precondición	[Condiciones sobre el estado del sistema que deben cumplirse para iniciar el caso de uso]	
Postcondición	[Efectos inmediatos que tienen la ejecución del caso de uso sobre el estado del sistema]	
Descripción	[Descripción del caso de uso]	
Resumen	[Resumen de alto nivel del funcionamiento	

Curso Normal

Nro.	Ejecutor	Paso o Actividad
[Nro. de paso]	[Actor ejecutor o especifica si es el sistema o subsistema]	[Descripción del paso actividad ejecutado]
[Se describe el proceso o secuencia de pasos ejecutadas usando frases cortas] [Cada paso del proceso puede ser ejecutado por los Actores o por el sistema] [Se describe la secuencia de acciones realizadas por los actores y la secuencia de actividades realizada por el sistema como respuesta].		

Tabla 3 Ejemplo de caso de uso
Fuente: La oficina de proyectos en informática

Diagramas de casos de usos

"Este tipo de diagrama es utilizado para representar y documentar a grandes rasgos, cómo interactúa un usuario o actor con el sistema de información en desarrollo a través de las operaciones o casos de uso que puede realizar el actor contra el sistema." (López, 2015, p.64)

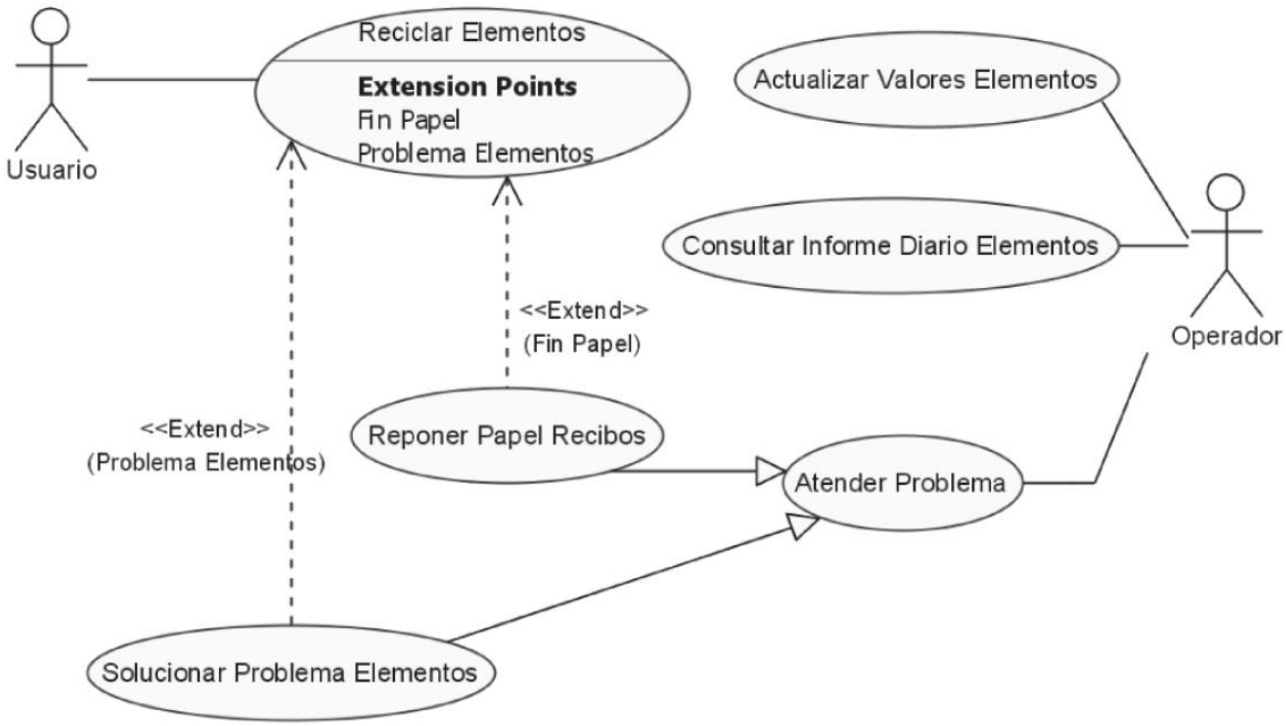


Figura 6 Ejemplo de diagrama de flujo
Fuente: Calidad de sistemas de información p.515

Cuando se habla de actores, se refiere a una "Entidad externa al sistema que participa en la secuencia de eventos del caso de uso. Puede ser una persona, un conjunto de personas, un sistema hardware, un sistema software o un reloj." (Teniente, 2015, p.69)

Diagramas de flujo

Un diagrama de flujo es un diagrama que describe un proceso, sistema o algoritmo informático. Se usan ampliamente en numerosos campos para documentar, estudiar, planificar, mejorar y comunicar procesos que suelen ser complejos en diagramas claros y fáciles de comprender. Los diagramas de flujo emplean rectángulos, óvalos, diamantes y otras numerosas figuras para definir el tipo de paso, junto con flechas conectoras que establecen el flujo y la secuencia. Pueden variar desde diagramas simples y dibujados a mano hasta diagramas exhaustivos creados por computadora que describen múltiples pasos y rutas. (Lucidchart, s.f.)

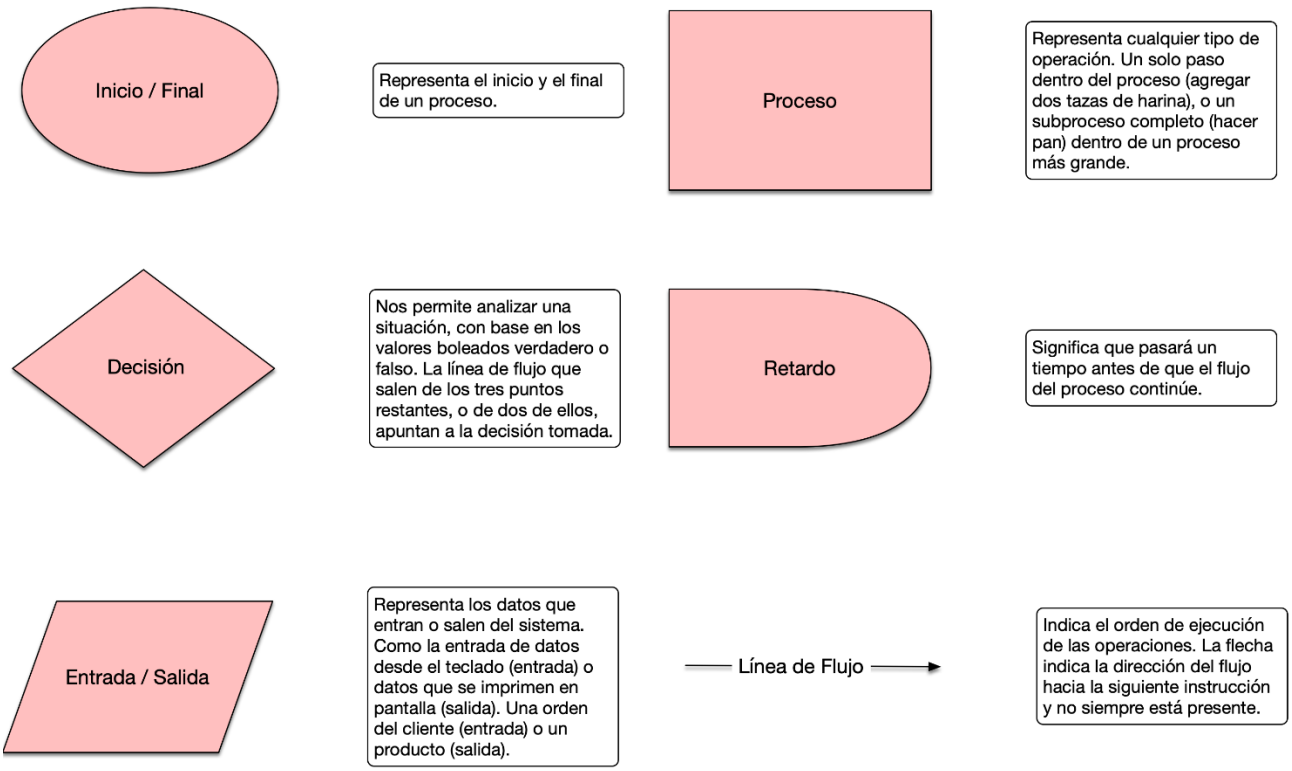


Figura 7 Figuras de diagramas de flujo
Fuente: KodigoSwift.com

Lenguaje UML

"UML es un lenguaje gráfico diseñado para especificar, visualizar, modificar, construir y documentar un sistema. Permite una visualización estándar de diferentes artefactos, entre otros, actividades, actores, lógicas de negocio y esquemas de bases de datos." (Casas y Caralt, 2014, p.47)

Requerimientos o requisitos del software

Los requerimientos del sistema se obtienen dentro del proceso de análisis, en esta parte se debe de enlistar todos "los requisitos funcionales, que corresponden a aquellas capacidades que describen las funciones que debe cumplir el software, y los requisitos no funcionales que son los que se refieren a la calidad" (Otárola, 2018, p.22)

2.9.1.2 Diseño

> En esta etapa se pretende determinar el funcionamiento de una forma global y general, sin entrar en detalles. Uno de los objetivos principales es establecer las consideraciones de los recursos del sistema, tanto físicos como lógicos. Se define por tanto el entorno que requerirá el sistema, aunque también se puede establecer en sentido contrario, es decir, diseñar el sistema en función de los recursos de los que se dispone. (Casado, 2015, p.18)

Prototipo

"Un prototipo es una versión inicial de un sistema de software que se usa para demostrar conceptos, tratar opciones de diseño y encontrar más sobre el problema y sus posibles soluciones" (Sommerville, 2011, p.45)

2.9.1.3 Programación o codificación

En esta etapa:

> El programador contará con un análisis completo del sistema que hay que codificar y con una especificación de la estructura básica que se necesitará, por lo que en un principio solo habría que traducir el cuaderno de carga en el lenguaje deseado para culminar la etapa de codificación, pero esto no es siempre así, las dificultades son recurrentes mientras se modifica. Por supuesto que cuanto más exhaustivo haya sido el análisis y el diseño, la tarea será más sencilla, pero nunca está exento de necesitar un reanálisis o un rediseño al encontrar un problema al programar el software. (Casado, 2015, p.19)

2.9.1.4 Pruebas

En esta etapa del ciclo de la programación, lo que se busca es:

> Mejorar la calidad detectando defectos y problemas, mediante la revisión del comportamiento del programa de manera dinámica, por medio de una cantidad finita de casos-prueba, seleccionados con relación al comportamiento esperado; se eligen según criterios que determinan si son o no adecuadas para el software a evaluar, se analizan respecto a las técnicas de gestión de calidad, y se clasifican en: pruebas de validez y verificación, de depuración, de programación y de certificación. (Otálora, Callejas y Alarcón, 2018, p.26)

2.9.1.5 Mantenimiento

Según el autor Pérez; “Se conoce como plan de mantenimiento de software a las directrices y procedimientos que se deben seguir para prevenir o solucionar cualquier fallo o avería en el software de un equipo informático. (Pérez, 2016, p.16)

2.9.2 Metodologías para el desarrollo de software

El propósito de la metodología es ofrecer unas guías para la gestión y desarrollo de software en un entorno distribuido utilizando las ventajas proporcionadas por la integración de métodos ágiles como Scrum y metodologías tradicionales como el Proceso Unificado de Desarrollo. (Piattini, Vizcaíno y García, 2014, p.64)

2.9.2.1 Metodologías tradicionales

Modelo de cascada

El modelo en cascada es el primer ciclo de vida del software. Fue definido por Winston Royce a finales de 1970. Este es el más básico de todos los modelos. Su visión dice que el desarrollo de software es a través de una secuencia simple de fases. (López, 2015, p.19)

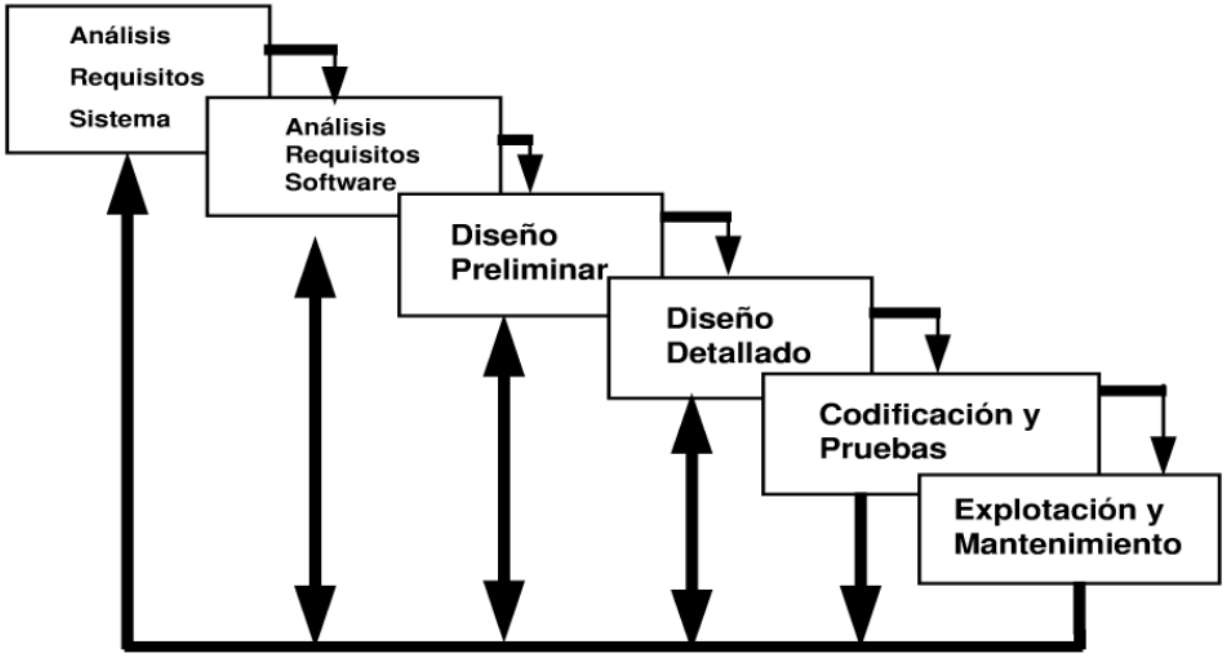


Figura 8 Diagrama del modelo de cascada
Fuente: Calidad de sistemas de información p.296

Modelo de espiral

El proceso de software se representa como una espiral, y no como una secuencia de actividades con cierto retroceso de una actividad a otra. Cada ciclo en la espiral representa una fase del proceso de software. Por ende, el ciclo más interno puede relacionarse con la factibilidad del sistema, el siguiente con la definición de requerimientos, el ciclo que sigue con el diseño del sistema, etcétera. (Sommerville, 2011, p.48)

Modelo incremental

El modelo incremental combina elementos del modelo en cascada con la filosofía interactiva de construcción de prototipos. Se basa en la filosofía de la construcción, incrementando a cada paso las funcionalidades del programa. Este modelo aplica secuencias lineales de forma escalonada, mientras progresa el tiempo en el calendario. Cada secuencia lineal produce un incremento del software. (López, 2015, p.21)

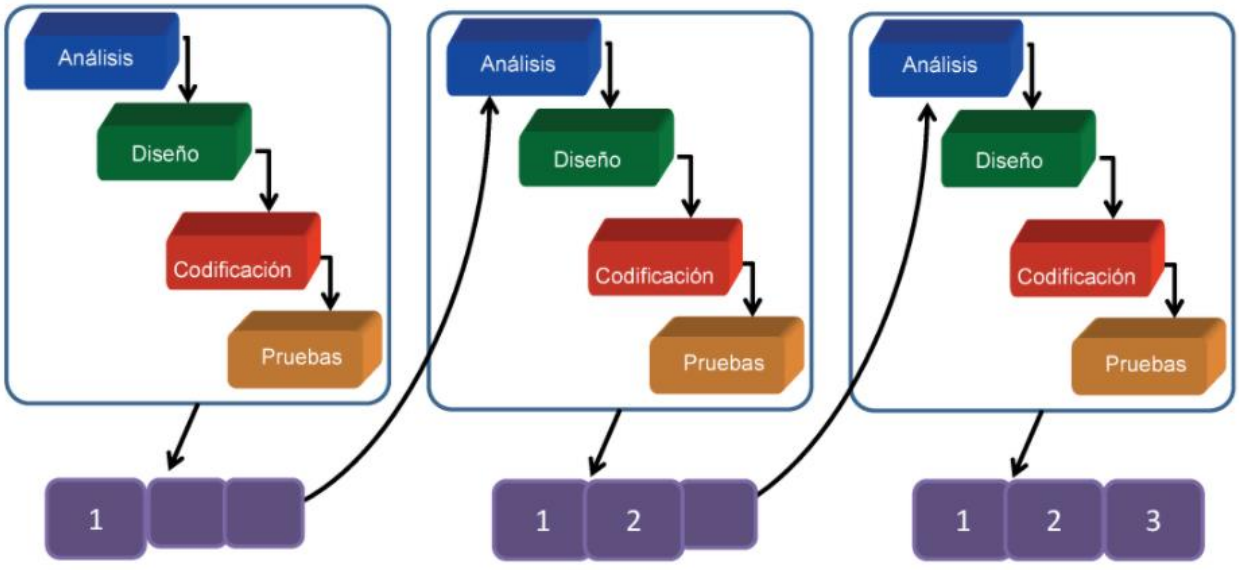


Figura 9 Diagrama de modelo incremental
Fuente: Programación web en el entorno de servidor p.21

Modelo V

El modelo en V es un proceso que representa la secuencia de pasos en el desarrollo del ciclo de vida de un proyecto. Describe las actividades y resultados que han de ser producidos durante el desarrollo del producto. La parte izquierda de la V representa la descomposición de los requisitos y la creación de las especificaciones del sistema. El lado derecho de la V representa la integración de partes y su verificación. V significa "Validación y Verificación". (López, 2015, P.22)

Figura 10 Diagrama de modelo V
Fuente: Programación web en el entorno de servidor p.23

2.9.2.2 Metodologías agiles

Las metodologías agiles fueron creadas aproximadamente en 1990, estas "apoyan universalmente en el enfoque incremental para la especificación, el desarrollo, y la entrega del software. Son más adecuados para el diseño de aplicaciones en que los requerimientos del sistema cambian, por lo general, rápidamente durante el proceso de desarrollo" (Sommerville, 2011, p.59)

Según los autores (Piattini y García, 2018, p.312) las metodologías agiles funcionan bajo los siguientes principios:

- Valorar más a los individuos y su interacción que a los procesos y las herramientas
- Valorar más el software que funciona que la documentación exhaustiva
- Valorar más la colaboración con el cliente que la negociación contractual
- Valorar más la respuesta al cambio que el seguimiento de un plan

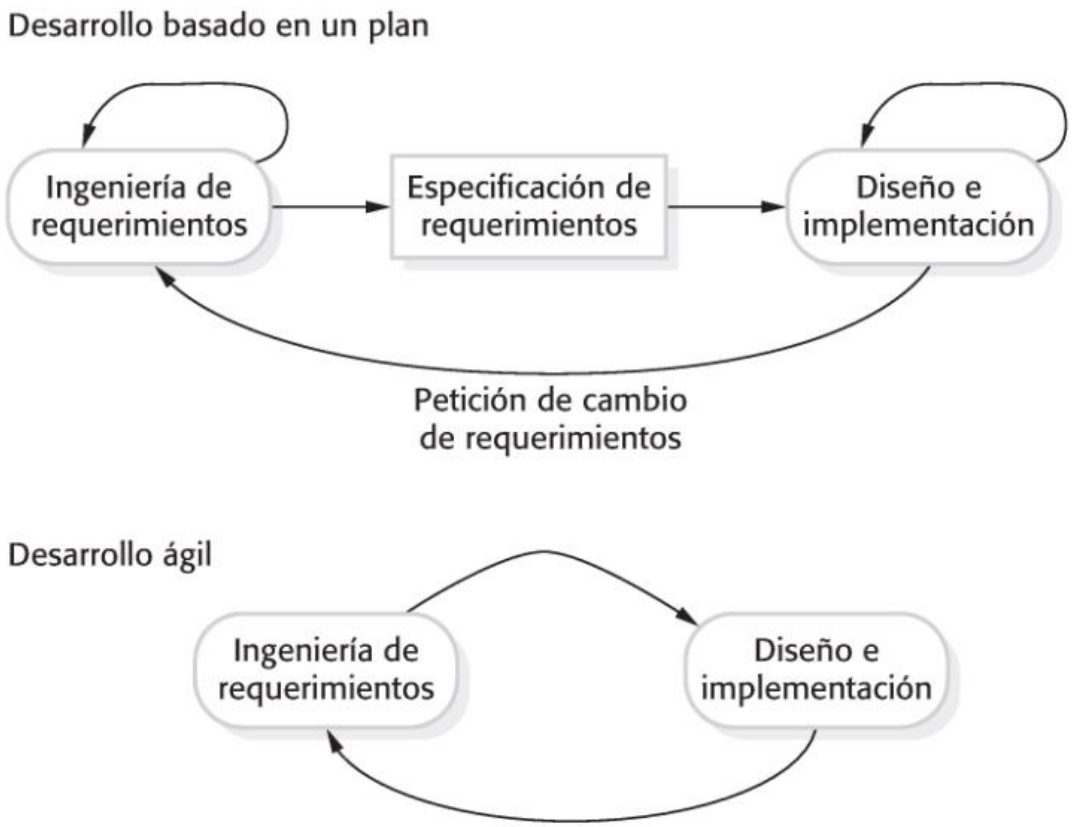


Figura 11 Desarrollo tradicional vs desarrollo ágil
Fuente: Ingeniería de Software (9ª. edición)

Las metodologías agiles más comunes son:

- Scrum
- Kanban
- XP

Scrum

Scrum es:

> La metodología ágil más popular en la actualidad, cada iteración se denomina “sprint”, durante la cual se implementarán características que provienen del “product backlog”. El equipo de desarrollo selecciona las historias de usuario que se van a desarrollar en el sprint conformando el “sprint backlog”, cuya composición es decidida por el equipo de desarrollo. (Piattini y García, 2015, p.314)

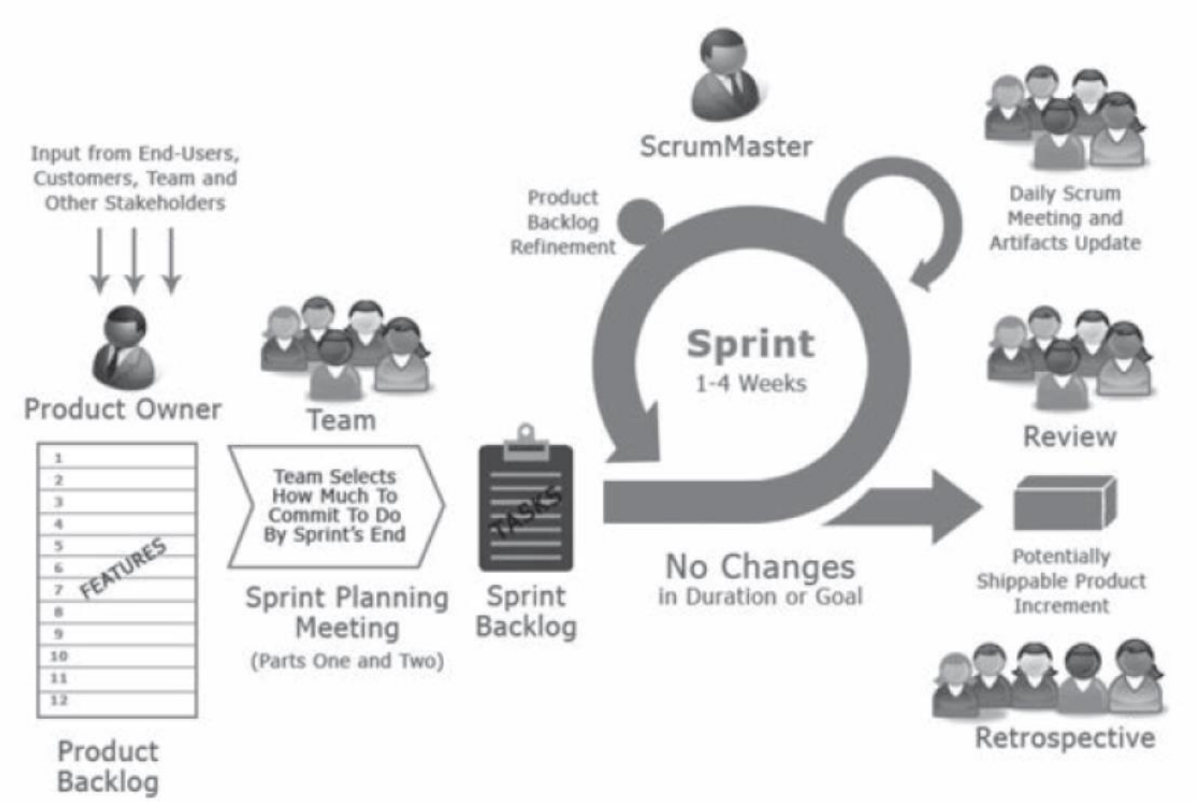


Figura 12 Diagrama de ciclo de vida Scrum
Fuente: Implantar Scrum con éxito p.49

Según el autor (Monte, 2016, p.21) Scrum trabaja bajo los siguientes pilares:

- Transparencia: los aspectos significativos del proceso tienen que ser conocidos por todo aquel que participa, lo cual conlleva que estos aspectos estén definidos mediante un

estándar común, de forma que todo el mundo tenga la misma percepción de las características de cada aspecto (por ejemplo, la definición de acabado).

- Inspección: todo proceso persigue un objetivo y, para llegar a ese objetivo, hace falta que los participantes en el proceso evalúen de manera continua sus resultados, y el proceso mismo, para detectar posibles desviaciones tan pronto como sea posible.
- Adaptación: cuando se detecta una desviación, la respuesta debe ser la adaptación; es decir, la adopción de acciones o planes que, o bien ayuden a corregir la desviación, o bien reconfiguren el objetivo.

Roles de Scrum

Scrum propone los siguientes roles para el funcionamiento de esta metodología:

Product Owner: "Una persona orientada al negocio cuyo objetivo sea maximizar el valor del negocio del producto" (Rad y Turley, 2019, p.21)

Stakeholder: "Los Stakeholder son los receptores del producto acabado y, por lo tanto, son quienes hacen la aceptación. Para hacer esto, están obligados a asistir a los sprint reviews" (Monte, 2016, p.54)

Scrum Master: "Esta persona asegura de que el marco de Scrum se sigue del todo y de forma correcta, lo cual requiere coaching, formación y capacidad de resolución de problemas" (Rad y Turley, 2019, p.19)

Development Team: Son "las personas que integran un equipo de desarrollo realizan diferentes tareas (estimación, diseño, desarrollo, pruebas, etc.), sin que exista una especialización (García y Piattini, 2019, p.322)

Herramientas de Scrum

Product Backlog

“El product backlog es la lista de funcionalidades, productos o acciones que conforman el producto que se ha de construir. El product backlog se escribe en «el idioma» del cliente y se compone de user stories (historias de usuario)” (Monte, 2016, p.55)

Task Board (Panel de tareas)

Este panel muestra las tareas que tienen asignadas el Development Team. Normalmente esta tabla está compuesta de tres columnas donde presentan el estado de cada actividad:

- To do (Por hacer)
- In progress (En progreso)
- Completed (Terminado)

Adicionalmente se puede incluir columnas según la necesidad del Development Team, como; pendiente, pendiente de aceptación, en pruebas entre otros.

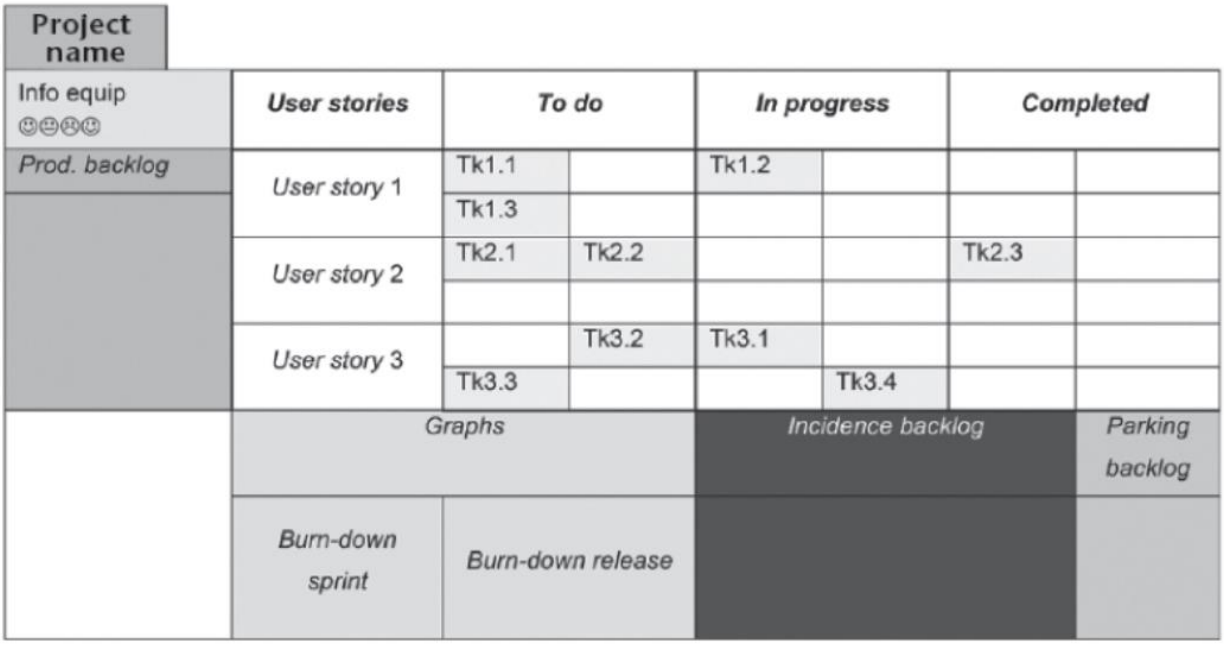


Tabla 4 Ejemplo de tablero de Scrum
Fuente: Implantar Scrum con éxito p.71

Sprint

Un sprint es la unidad de tiempo que determina un ciclo de desarrollo con Scrum. En este tiempo deben llevarse a cabo obligatoriamente las actividades siguientes:

- El sprint planning.
- Una reunión diaria (daily meeting) con el DT.
- Tantas reuniones de refinamiento (grooming) como sean necesarias para resolver cuestiones del sprint y para preparar el sprint siguiente.
- El sprint review para entregar el producto acabado.
- El sprint retrospective, para favorecer la mejora continua del equipo. (Monte, 2016, p.77)

Diferencias del modelo tradicional vs Scrum

Modelo tradicional (cascada)	Scrum
• Modelo predictivo • *Relay race* (carrera de relevos): para empezar una fase hay que acabar la anterior • Organizado jerárquicamente • Departamental • Objetivos completos • Controlado en tiempo, presupuesto, alcance y calidad	• Modelo adaptativo • Holístico*. Deporte de equipo • Aproximación matricial: diversas • tareas pueden estar ejecutándose • a la vez. La responsabilidad de las tareas es compartida por todos los miembros del equipo • Autogestionado • Entregas incrementales, • aportación continúa de valor • Controlado en tiempo, presupuesto, • alcance, calidad y expectativas (el cliente colabora)

Tabla 5 Diferencias de Scrum vs el modelo tradicional
Fuente: Implantar Scrum con éxito p.28

Kanban

> Kanban es un sistema para mostrar el estado de las tareas de un proyecto. Ideado por el industrial Taiichi Ohno en 1953. Kanban es la unión de dos conceptos: kan, que significa 'visual', y ban, que significa 'tarjeta'. Tarjetas visuales situadas en un tablero y que indican en todo momento el estado de fabricación de una pieza, un pro- ducto o cualquier elemento que requiere de un proceso de acciones para su construcción. (Monte, 2016, p.43)

En otras palabras, Kanban es un tablero donde muestra el seguimiento de las diferentes tareas del desarrollo que muestra el estado actual de cada tarea, para ello normalmente se utilizan las siguientes columnas:

- To Do (Por hacer)
- Doing (Haciendo)
- Test (Pruebas)
- Deployment (Despliegue)
- Done (Realizado)

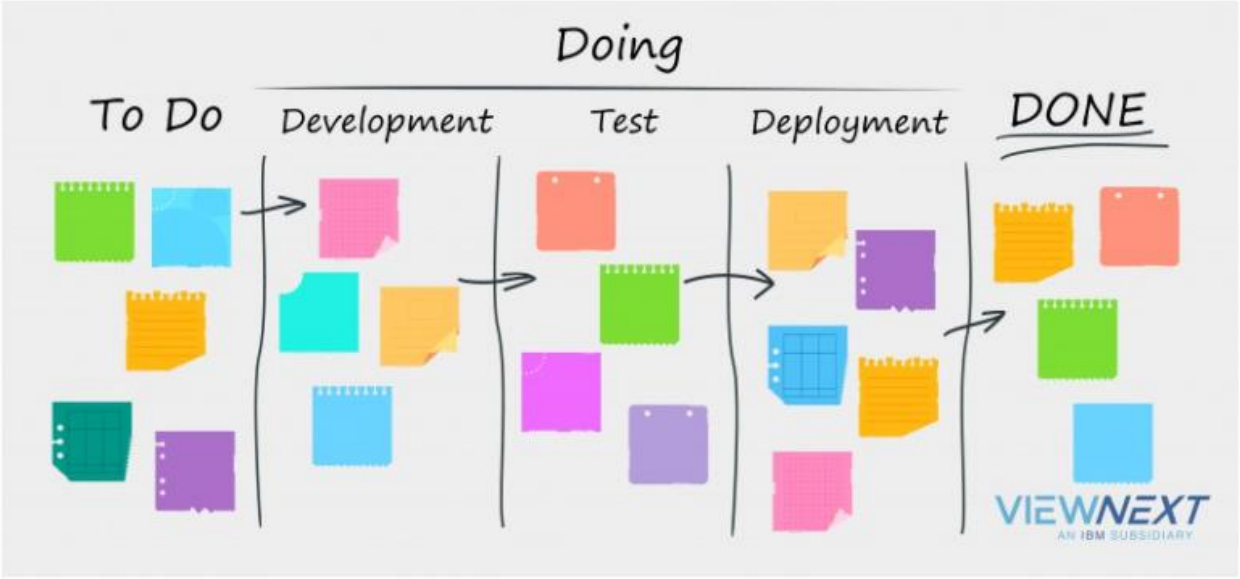


Figura 13 Ejemplo de Kanban
Fuente: viewnext.com

Metodología XP

Según el autor Sommerville “El nombre lo acuño Beck (2000) debido a que el enfoque de desarrolló llevando a niveles “extremos” las practicas reconocidas, como el desarrollo iterativo” (2011, p.64) donde su “objetivo principal de esta técnica es la de proporcionar al equipo de desarrolladores herramientas para adaptarse a los cambios, por encima de la previsibilidad completa que es habitual en metodologías tradicionales.” (Monte, 2016, p.40)

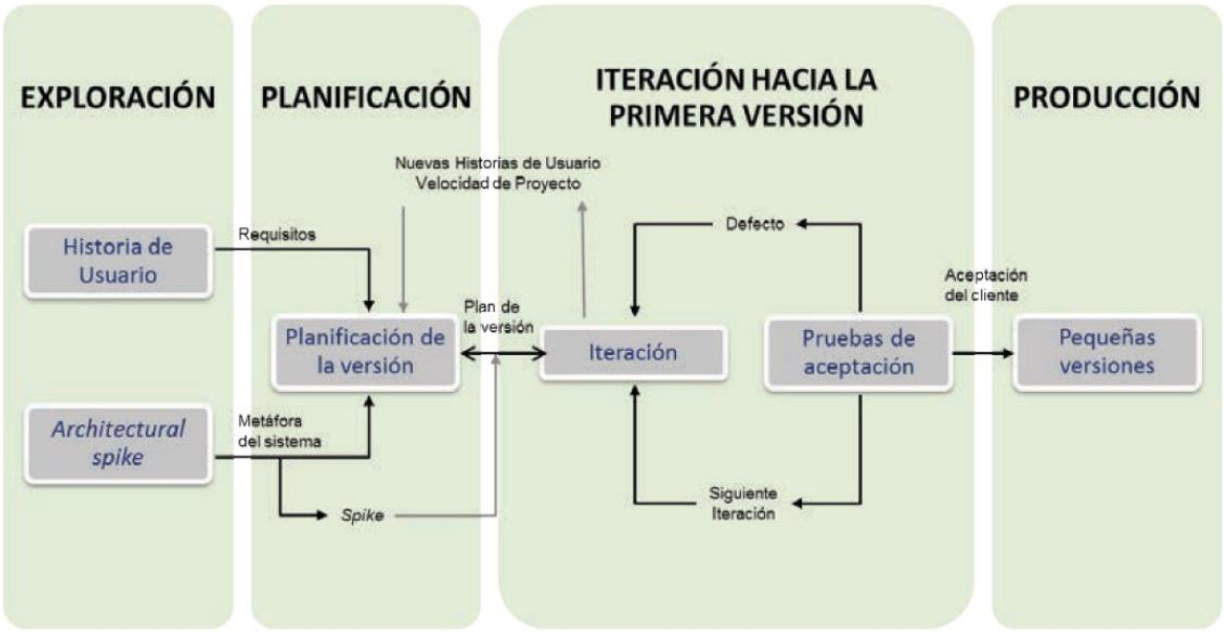


Figura 14 Ciclo de vida de Extreme programming
Fuente: Calidad de sistemas de información p.321

Las fases de esta metodología son:

Exploración

> Fase en la que los clientes plantean las necesidades a partir de historias de usuario que serán realizadas durante la primera iteración. A su vez el equipo de desarrollo revisa las tecnologías, prácticas y herramientas a ser utilizadas durante el proyecto. (García y Piattini, 2019, p.320)

Planificación

Durante esta etapa se sacan las prioridades según las historias de usuarios para poder iniciar las iteraciones hasta lograr la primera versión del software.

Desarrollo o iteración hacia la primera versión

> En cada iteración el cliente decide las historias de usuario que se realizarán, que serán los requisitos de la siguiente versión. Al finalizar cada iteración, el cliente realizará las pruebas funcionales para asegurarse de que todo funciona correctamente. En caso de que existan defectos, estos se solucionarán en las siguientes iteraciones. Por lo tanto, al finalizar la última iteración, el sistema estará en producción. (García y Piattini, 2019, p.320)

Producción

"El pase de un sistema a producción requiere de pruebas de aceptación y comprobaciones adicionales. Y se tiene que decidir si se incluyen en la versión actual nuevas funcionalidades o modificaciones. (García y Piattini, 2019, p.320)

Principios de Extreme Programming

> Los principios nos permiten tomar mejores decisiones cuando nos encontramos con distintas alternativas. Es mejor elegir una alternativa que cumpla con los principios de forma directa que una que no lo haga de la misma forma. Cada principio encarna uno o más valores. Un valor puede ser vago y difícil de aplicar, por ejemplo, algo que es "Simple" para una persona, para otra puede ser complejo "Complejo". Un principio es más concreto, u obtienes feedback rápidamente o no. (López, 2020, párr.6)

Principio o práctica	Descripción
Planeación incremental	Los requerimientos se registran en tarjetas de historia (*story cards*) y las historias que se van a incluir en una liberación se determinan por el tiempo disponible y la prioridad relativa. Los desarrolladores desglosan dichas historias en "tareas" de desarrollo.
Liberaciones pequeñas	Al principio se desarrolla el conjunto mínimo de funcionalidad útil, que ofrece valor para el negocio. Las liberaciones del sistema son frecuentes y agregan incrementalmente funcionalidad a la primera liberación.
Diseño simple	Se realiza un diseño suficiente para cubrir sólo aquellos requerimientos actuales.
Desarrollo de la primera prueba	Se usa un marco de referencia de prueba de unidad automatizada al escribir las pruebas para una nueva pieza de funcionalidad, antes de que esta última se implemente.
Refactorización	Se espera que todos los desarrolladores refactoricen de manera continua el código y, tan pronto como sea posible, se encuentren mejoras de éste. Lo anterior conserva el código simple y mantenible.
Programación en pares	Los desarrolladores trabajan en pares, y cada uno comprueba el trabajo del otro; además, ofrecen apoyo para que se realice siempre un buen trabajo.
Propiedad colectiva	Los desarrolladores en pares laboran en todas las áreas del sistema, de manera que no se desarrollan islas de experiencia, ya que todos los desarrolladores se responsabilizan por todo el código. Cualquiera puede cambiar cualquier función.
Integración continua	Tan pronto como esté completa una tarea, se integra en todo el sistema. Después de tal integración, deben aprobarse todas las pruebas de unidad en el sistema.
Ritmo sustentable	Grandes cantidades de tiempo extra no se consideran aceptables, pues el efecto neto de este tiempo libre con frecuencia es reducir la calidad del código y la productividad de término medio.
Cliente en sitio	Un representante del usuario final del sistema (el cliente) tiene que disponer de tiempo completo para formar parte del equipo XP. En un proceso de programación extrema, el cliente es miembro del equipo de desarrollo y responsable de llevar los requerimientos del sistema al grupo para su implementación.

Tabla 6 Principios de XP programing
Fuente: Ingeniería del software 9a edición p.66

2.10 PAYPAL

PayPal es un servicio que ofrece medios de pago virtuales de forma segura a nivel mundial, este está presente en más de 200 países, donde por medio de la página https://www.paypal.com/cr/home se pueden realizar transferencias monetarias de cuenta a cuenta o realización de pagos donde acepta tarjetas de crédito o débito.

Además, PayPal cuenta con un ambiente de pruebas llamado Sand Box, este ambiente permite a los programadores crear cuentas de PayPal de prueba o tarjetas de débito/crédito para simular las compras a través de sus servicios, para poder ir mostrando el comportamiento que tiene la aplicación en desarrollo.

2.11 SERVICIO DE HOSTING

El servicio de hosting se “refiere al «espacio» donde se mantiene un sitio web, es decir, que un hosting sirve para alojar una web y que todo el mundo pueda verla en Internet a través del dominio que tenga” (Alonso, 2020)

2.11.1 Dominios y subdominios

El dominio es a lo que llamamos comúnmente como “la dirección de internet” y “está compuesto por un nombre y su extensión de dominio. El nombre de dominio es esencial para la identidad de cualquier marca, compañía, entidad legal o individuo que desee posicionarse en la red de Internet. (Mailify, s.f.)

El subdominio es una subclasificación del nombre del dominio principal, normalmente se utiliza para otros fines de segundo nivel administrativamente.

Figura 15 Ejemplo de dominio y subdominio
Fuente: Mailify ¿Qué es un dominio y un subdominio?

2.11.2 Certificado SSL

SSL (Secure Sockets Layer o capa de conexión segura) es un estándar de seguridad global que permite la transferencia de datos cifrados entre un navegador y un servidor web. Es utilizado por millones de empresas e individuos en línea a fin de disminuir el riesgo de robo y manipulación de información confidencial (como números de tarjetas de crédito, nombres de usuario, contraseñas, correos electrónicos, etc.) por parte de hackers y ladrones de identidades. Básicamente, la capa SSL permite que dos partes tengan una "conversación" privada. (Verisign, s.f.)

2.12 LIBRERÍAS

Es un conjunto de subprogramas que sirven para programar una aplicación. Las librerías contienen código y datos que pueden ser llamados desde otro programa principal. De esta forma se evita que cada programa principal desarrolle el código de la librería. En la inclusión del código de librería, éste pasa a formar parte del programa principal. (Vara Mesa, Verde Marín, López Sáenz, 2015, p.237)

2.12.1 Bootstrap

Bootstrap, es un framework originalmente creado por Twitter, que permite crear interfaces web con CSS y JavaScript, cuya particularidad es la de adaptar la interfaz del sitio web al tamaño del dispositivo en que se visualice. Es decir, el sitio web se adapta automáticamente al tamaño de una PC, una Tablet u otro dispositivo. Esta técnica de diseño y desarrollo se conoce como "responsive design" o diseño adaptativo. (Solís, 2014, párr.1)

2.12.2 Materialize

Materialize es una librería de CSS que contiene integración con JavaScript, que está enfocada en las animaciones y transiciones de las páginas web con un diseño adaptativo y de fácil realización.

2.12.3 jQuery

jQuery es una popular biblioteca de JavaScript. Fue creada por John Resig en el 2006 con el objetivo de facilitarle el uso de JavaScript en los sitios web a los desarrolladores. No es un lenguaje de programación separado y funciona en conjunto con JavaScript. (B, hostinger.es, 2019)

2.12.4 Vue.js

Vue.js es una librería basada en JavaScript que está enfocada en la interfaz del usuario, donde su principal funcionamiento es realizar validaciones, interacción con el usuario, realización peticiones http y conectividad con los controladores del framework.

2.12.5 Summernote

Summernote es una librería con licencia MIT de JavaScript que permite crear diferentes tipos de editores de texto con integración de Bootstrap bajo el esquema WYSIWYG, que en español significa “Lo que ves, es lo que tienes” logrando que el texto ingresado y diseñado por el usuario sea generado por medio de HTML con el CSS listo para ser visualizado.

2.13 WEBSERVICE (SERVICIOS WEB)

> Un servicio web es un conjunto de protocolos y estándares que permiten comunicar dos sistemas a través de una red. Habitualmente los servicios web actúan para intercambiar datos (comunicarse), entre dos aplicaciones. estas aplicaciones suelen estar desarrolladas en lenguajes de programación distintos. además, pueden estar en plataformas (sistemas operativos o arquitecturas) diferentes. es habitual, y el uso más generalizado que los servicios web, intercambien los datos a través de internet. aunque esta es la generalidad podrían implementarse servicios web en otro tipo de red. (Vara Mesa, Verde Marín, López Sáenz, 2015, p.197)

CAPÍTULO III: MARCO METODOLÓGICO

3.1 TIPO Y ENFOQUE DE LA INVESTIGACIÓN

3.1.1 Tipo de la investigación

La investigación presente pretende desarrollar una herramienta web que tiene como objetivo principal mejorar los procesos de recaudación de los fondos monetarios de una ONG, para ello el tipo de investigación es la aplicada, porque según el autor Baena:

> La investigación aplicada tiene como objeto el estudio de un problema destinado a la acción. La investigación aplicada puede aportar hechos nuevos… si proyectamos suficientemente bien nuestra investigación aplicada, de modo que podamos confiar en los hechos puestos al descubierto, la nueva información puede ser útil y estimable para la teoría. (Baena, 2014, p.22)

Por lo anterior, se busca obtener los conocimientos técnicos y estructurales de la ONG por medio de fuentes primarias para que la aplicación de la investigación aplicada brinde una solución eficiente para el problema que actualmente tiene.

3.1.2 Enfoque de la investigación

El enfoque de la investigación es de tipo cualitativo, ya que de acuerdo con Guerrero (2015) “los investigadores cualitativos hacen registros narrativos de los fenómenos que son estudiados mediante técnicas como la observación participante y las entrevistas no estructuradas o en profundidad.” (p.58), además que el presente proyecto cumple con el proceso de una investigación cualitativa, donde este proceso:

> Se desenvuelve en cinco fases de trabajo: 1. Definición del problema, 2. Diseño del trabajo, 3. Recogida de datos, 4. Análisis de los datos, 5. Validación e informe. Cada una de las técnicas principales cualitativas (la observación participante, la entrevista personal, la historia de vida, el estudio de casos…) imprime un sello particular a cada una de las cinco fases, lo mismo que lo hacen el experimento o el survey de masas. Aun así, es posible establecer un estilo cualitativo propio como resultado de aplicar a todo el proceso,

> en cada una de sus fases, una serie de “criterios” o principios orientadores (Pérez, Pérez y Seca, 2020, p.170)

3.2 FUENTES DE INFORMACIÓN

3.2.1 Fuentes primarias

Las fuentes primarias según Martínez (2012) es “la información de primera mano – o datos primarios – es aquella que ha sido obtenida, organizada y formulada por el propio investigador” (p.134) por lo que la principal fuente de información del proyecto es la entrevista realizada al investigador científico de Namá Conservation Juan Carlos Cruz junto con el departamento de inteligencia comercial; logrando como resultado, captar el proceso actual por el que la ONG puede recibir las donaciones, para con ello, poder levantar los requerimientos y así mismo realizar las limitaciones y alcances del proyecto.

3.2.2 Fuentes secundarias

> Son documentos que compilan y reseñan la información publicada en las fuentes primarias. Retoman los documentos primarios u originales. Proporcionan una síntesis de la información que existe en los documentos primarios sobre temas de interés; además, se utilizan para remitir a los usuarios a documentos cuyos contenidos puedan ayudar a solucionar sus necesidades de información. (Del Castillo y Olivares, 2014, p.133)

Para efectos del proyecto, la información secundaria se obtiene por medio de libros con temas relacionados, documentación sobre investigaciones y sitios web que brinden aporte para la investigación.

3.2.3 Sujeto de información

Los sujetos de información son las personas que son parte de Namá Conservation a quienes se entrevistaron dentro de las reuniones para el desarrollo del proyecto. Ver tabla 7.

Puesto	Nombre	Relación con el tema
Investigador científico de campo	Juan Carlos Cruz	Alta
Inteligencia comercial	Carla Galván	Alta
Inteligencia comercial	Sebastián Miranda Salazar	Alta
Miembro de la junta directiva	Giovanni Miranda Camacho	Alta

Tabla 7 Sujetos de información
Fuente: Elaboración propia

3.3 TÉCNICAS Y HERRAMIENTAS DE RECOLECCIÓN DE DATOS

3.3.1 Observación

La observación es “uno de los primeros métodos científicos utilizados en la investigación y se utiliza para la obtención de información primaria acerca de los objetos investigados o para la comprobación empírica de las hipótesis. La observación científica es sistemática, consciente y objetiva.” (Fresno, 2019, p.114)

La técnica de la observación se utilizó para visualizar los diferentes procesos sobre la gestión de la recaudación de donaciones y registro de voluntariado; con el objetivo de crear el análisis para obtener los requerimientos funcionales y casos de uso que se deben de contemplar para el desarrollo de la aplicación web.

3.3.2 Entrevista

> Es el medio que permite la obtención de información de fuente primaria, amplia y abierta, en dependencia de la relación entrevistador entrevistado. Para ello es necesario que el entrevistador tenga definido claramente los objetivos de la entrevista y cuáles son los aspectos relevantes sobre los que se pretende obtener información. (Fresno, 2019, p.115)

La entrevista es la principal técnica de recolección de datos durante la presente investigación, debido a que se realizaran entrevistas a las partes más involucradas del proyecto, para poder obtener la información relevante para el desarrollo del proyecto.

3.3.3 Reuniones

Por medio de las reuniones, se podrán presentar todos los avances y revisiones de la aplicación web según los requerimientos obtenidos por medio de las herramientas de recolección de datos; el personal presente de Namá Conservation brindara retroalimentación sobre el trabajo realizado y en caso de que se solicite algún cambio de mejora en la aplicación web, serán analizados para su previa evaluación, con estas reuniones se pretende satisfacer las necesidades de Namá Conservation.

3.4 VARIABLES DE LA INVESTIGACIÓN

Objetivos específicos	Variables asociadas	Descripción
Analizar e identificar el proceso actual de la organización con respecto a la gestión para recibir donaciones y solicitudes de voluntariados, mediante técnicas o herramientas de recolección de datos.	Análisis sobre el diagnostico actual	Comprender el proceso de la ONG para recibir donaciones monetarias y solicitudes de voluntariados para que el software desarrollado pueda mejorar el proceso existente.

Identificar los requerimientos funcionales y no funcionales junto con las necesidades que se deben contemplar en el proyecto.	Requerimientos funcionales y no funcionales	Definir los requerimientos necesarios para poder desarrollar el software
Diseñar los diagramas de base de datos y los diagramas de flujo, del nuevo proceso de recolección de donativos y solicitudes de voluntariados que permita cubrir las necesidades plasmadas en los requerimientos mediante lenguaje UML.	Diseño del software Diagramas de flujo Diseño de base de datos	Elaboración de diagramas de flujo y de bases de datos para el software
Desarrollar el sistema web para mejorar el proceso de recaudación de donaciones, publicaciones y solicitudes de voluntariados bajo el patrón de arquitectura MVC..	Desarrollo del software	Realización del software tomando en cuenta el análisis, requerimientos y diagramas de flujo
Implementar la aplicación web desarrollada en el hosting, con el fin de poner en marcha la solución del problema principal.	Implementación del software	Ejecutar el software en el internet para poder recibir las donaciones monetarias y las solicitudes de voluntariados

Tabla 8 Variables de la investigación
Fuente: Elaboración propia

3.5 DISEÑO DE LA INVESTIGACIÓN

El diseño de la investigación se definió en las siguientes etapas:

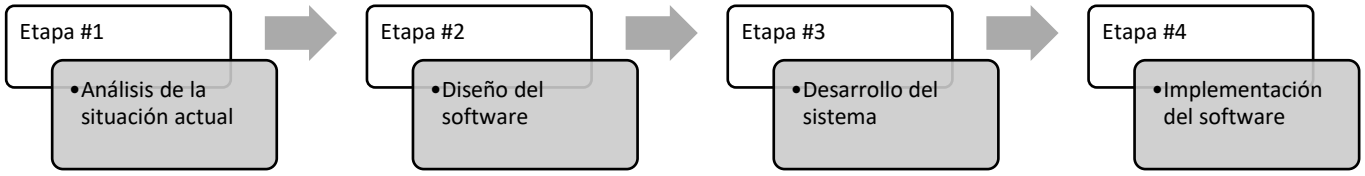


Figura 16 Diagrama de diseño de investigación
Fuente: Elaboración propia

3.5.1 Etapa #1: Análisis de la situación actual

Durante esta etapa se analiza las necesidades actuales que presenta la ONG para determinar la situación presente de los procesos para recaudar las donaciones monetarias junto con el proceso de registro de voluntariado. Para ello se aplican las técnicas de la entrevista, observación y reuniones, para poder obtener la información base para el levantamiento de los requerimientos del sistema.

3.5.2 Etapa #2: Diseño del software

En la segunda etapa de la investigación, se elaboran los listados de los requerimientos funcionales y no funcionales que deben de ser parte del software junto con su respectiva documentación. Además, se diseñan los diagramas para mostrar el comportamiento que la aplicación debe tener, para ello se utilizan como herramientas los casos de usos, diagramas de flujo, diagramas de bases de datos entre otros.

3.5.3 Etapa #3: Desarrollo del sistema

Una vez finalizada la etapa #2 y teniendo toda la documentación en mano, se procederá a desarrollar la aplicación, tomando en cuenta los requerimientos que debe de tener el sistema. Por lo tanto, se utilizará la herramienta Visual Studio Code para poder realizar la programación del software.

3.5.4 Etapa #4: Implementación del software

En la última etapa se realizará la implementación de la aplicación web en el servidor, donde por medio de una reunión se presentará el proyecto para su previa aprobación y una vez que esté aprobado la investigación se dará por concluida.

3.6 MATRIZ DE COHERENCIA

Objetivo	Entregables	Fase o etapa	Técnica de recolección de datos	Instrumentos	Temas relacionados para el marco teórico
Analizar e identificar el proceso actual de la organización con respecto a la gestión para recibir donaciones y solicitudes de voluntariados, mediante técnicas o herramientas de recolección de datos	Documento descriptivo con el proceso actual para la recaudación de donaciones y solicitudes de voluntariados	Etapa #1	Entrevista	Reunión	Ciclo de programación, Análisis
Identificar los requerimientos funcionales y no funcionales junto con las necesidades que se deben contemplar en el proyecto	Documento descriptivo con los requerimientos funcionales y no funcionales del software	Etapa #1	Entrevista	Reunión	Casos de uso, historias de usuario, requerimientos funcionales y no funcionales
Diseñar los diagramas de base de datos y los diagramas de casos de uso, del nuevo proceso de recolección de donativos y solicitudes de voluntariados que permita cubrir las	Diagramas de flujo y los diagramas de bases de datos	Etapa #2	Entrevista, observación	Reunión	Diagrama de casos de uso, bases de datos, lenguaje UML

necesidades plasmadas en los requerimientos mediante lenguaje UML					
Desarrollar el sistema web para mejorar el proceso de recaudación de donaciones, publicaciones y solicitudes de voluntariados bajo el patrón de arquitectura MVC	Manual de usuario del software	Etapa #3	Observación	Reunión	Software, ingeniería del software, lenguaje de programación, programación orientada a objetos, programación web, HTML, JavaScript, CSS, PHP, Bases de datos, MYSQL, Framework, Laravel, metodología ágil
Implementar la aplicación web desarrollada en el hosting, con el fin de poner en marcha la solución del problema principal	Software implementado en la web	Etapa #4	Observación	Reunión	Pruebas, mantenimiento, producción

Tabla 9 Matriz de coherencia
Fuente: Elaboración propia

CAPÍTULO IV: DIAGNOSTICO DE LA SITUACIÓN ACTUAL

EL objetivo de este capítulo es dar a conocer la situación actual de la ONG Namá Conservation en el proceso de recolección de donaciones monetarias y registro de personas que desean realizar voluntariados. A continuación, se muestran los procesos actuales para la recolección de datos.

4.1 DIAGNOSTICO ADMINISTRATIVO U OPERATIVO

Actualmente, la ONG Namá Conservation, no cuenta con una página web informativa para presentar la información del conservatorio, sus proyectos u otros programas, por lo tanto, acude al uso de redes sociales para dar a conocer la organización.

Dentro de las redes sociales (en el segmento informativo de la ONG) Namá Conservation cuenta con un link para que el internauta sea redirigido a cuestionario en Google forms, donde una vez dentro del mismo, el usuario puede ingresar la información ya sea para realizar una donación o para registrarse como voluntariado. Ver Anexo C.

A continuación, se describe los procesos para la recaudación de donaciones y registro de voluntariados:

Recaudación de donaciones

Una vez que el internauta ha llenado el cuestionario de Google forms, este es recibido por personal de Namá Conservation y se realizan los siguientes pasos para la realización de la donación:

- Paso #1: Si el número telefónico ingresado previamente pertenece a Costa Rica el primer contacto seria realizado por medio de una llamada telefónica, en caso contrario, el contacto lo trataran de realizar por medio del correo que el internauta ingreso previamente.
- Paso #2: Una vez que se establece el contacto, el personal de Namá Conservation pone a disposición los medios por los cuales el internauta puede realizar la donación. Los medios disponibles para el recibimiento de donaciones son por transferencia a una cuenta del

banco Nacional de Costa Rica o por medio de la plataforma de PayPal, que en su mayoría es utilizada para recibir las donaciones que vienen del extranjero.

- Paso #3: Una vez que se establece la comunicación, Namá queda a la espera de que la persona quien realizo el contacto realice la donación. Ver figura 17.

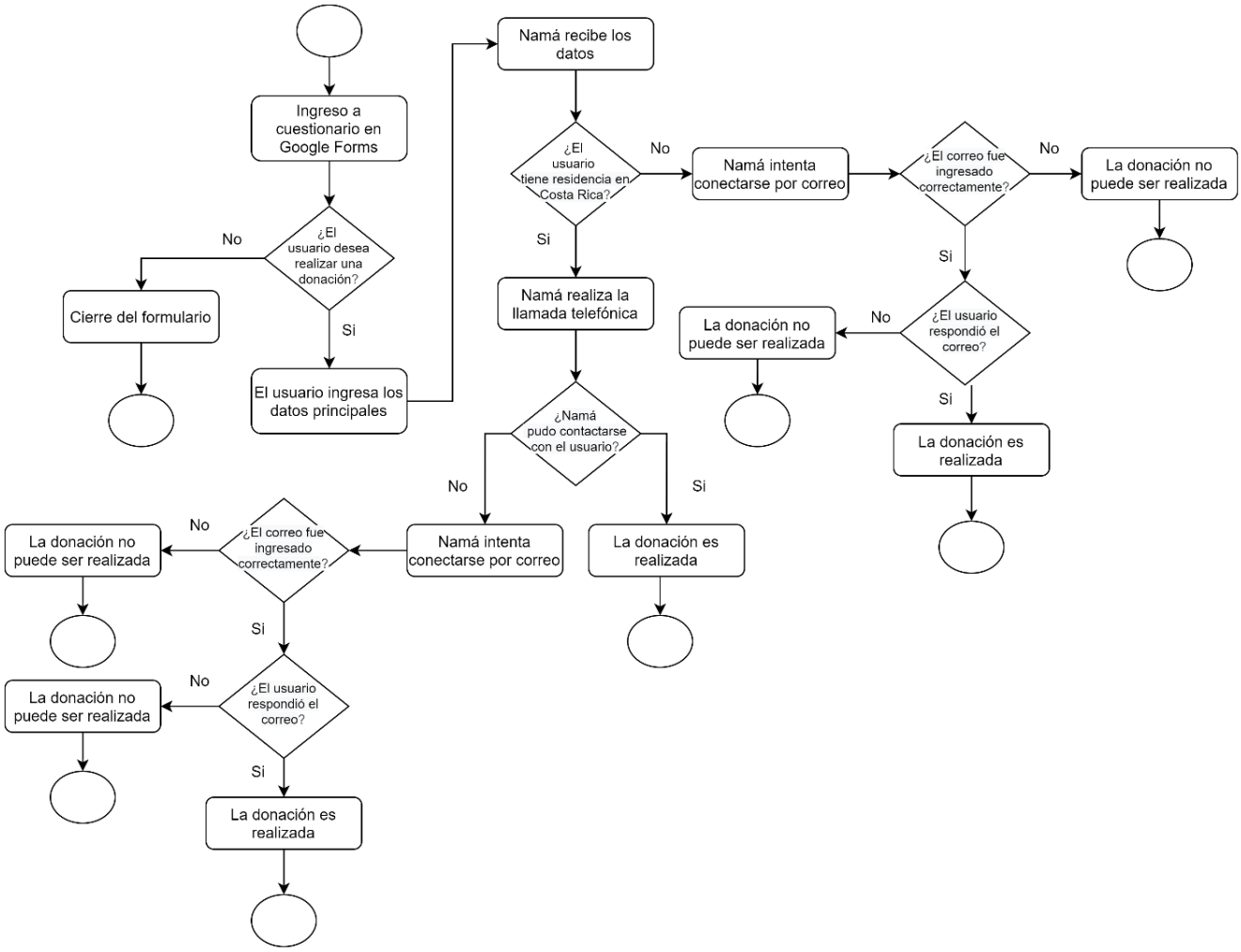


Figura 17 Diagrama de proceso de recaudación de donaciones
Fuente: Elaboración propia

Adicionalmente del proceso de recaudación de donaciones anterior, Namá Conservation también realiza campañas con los partners comerciales. Estas campañas normalmente son lideradas por los mismos partners y son enfocadas en realizar cenas, subastas o venta de artículos donados, donde cobran un monto de entrada al público general. Una vez finalizado el evento, los montos

recolectados por entradas son depositados directamente a las cuentas de Namá Conservation en el banco nacional de Costa Rica.

Inscripción de voluntariados

Cuando Namá Conservation obtiene los datos de las personas que desean realizar voluntariados, se realizan los siguientes pasos para la inscripción:

- Paso #1: La ONG trata de realizar el contacto, inicialmente se realiza por medio de una llamada telefónica o por correo electrónico.
- Paso #2: Una vez que se establece el contacto, se realiza una entrevista para poder recabar la información acerca de la persona y la disponibilidad de esta, para que pueda realizar un voluntariado según los programas disponibles. Ver figura 18.

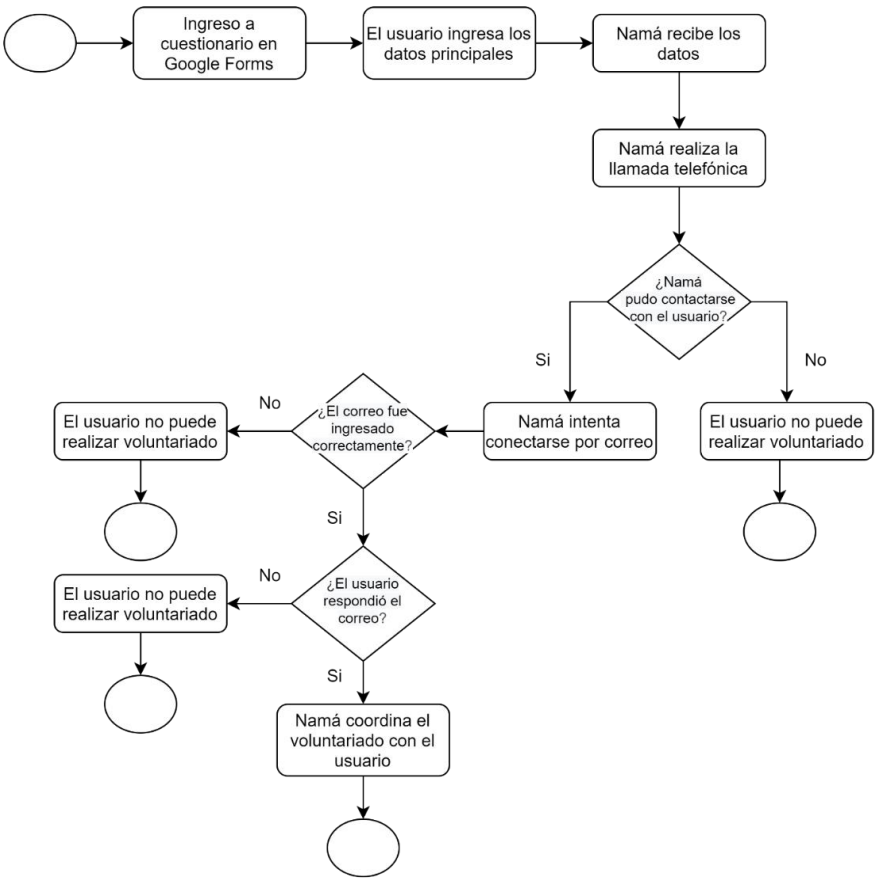


Figura 18 Diagrama de proceso de voluntariado
Fuente: Elaboración propia

4.2 DIAGNOSTICO TÉCNICO

4.2.1 Servicio de hosting

Actualmente, la ONG Namá Conservation cuenta con un servicio de hosting con el proveedor Hostgator, donde tiene comprado el dominio: namaconservation.org junto con los siguientes servicios:

- Servicios de correo electrónico ilimitado.
- Creación ilimitada de subdominios bajo el dominio namaconservation.org
- Acceso a panel del control del alojamiento web.
- Alojamiento de páginas web.
- Herramientas de estadísticas para el rendimiento o tráfico de las páginas web.
- Instaladores de aplicaciones web.
- Directorios protegidos por contraseña.
- Herramientas de gestión expertas.
- Certificado SSL

4.2.2 Servidor del hosting

El servidor del servicio de hosting cuenta con las siguientes características:

- Procesador Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz de 20 cores
- 60 GB de memoria RAM
- Disco duro de 6x2TB (RAID 6)
- Sistema operativo CentOS 7.6 Enterprise Linux x86

El servidor descrito anteriormente, cuenta con una velocidad de conexión a internet de 100mbps (megabits por segundo)

4.3 DIAGNÓSTICO DE PERCEPCIÓN

Para poder brindar el diagnostico de percepción, se realizó un análisis con base en la perspectiva de las personas que son parte de la organización de Namá Conservation, esto fue captado por medio de reuniones donde se utilizó la herramienta de la entrevista, con el fin de comprender los procedimientos actuales para que con ellos se pueda obtener y desarrollar una aplicación web que permita darle una mejora a la organización. A continuación, se presenta un análisis de las preguntas realizadas en la entrevista.

4.3.1 Análisis de la entrevista

La entrevista se realizó principalmente al investigador científico Juan Carlos Cruz de Nama Conservation junto a el equipo de inteligencia comercial, quienes son parte esencial en la estructura de los procesos de la organización. A continuación, se destacan los principales inconvenientes relacionados a las preguntas expuestas.

1) ¿Cuál es el proceso actual de la recaudación de donaciones?
    - Existe un alto porcentaje de donaciones que no llegan a efectuarse por errores del usuario al completar el formulario de contacto o porque no se llega a establecer la comunicación, ya sea por diferencias horarias o por falta de respuesta del donador.
    - Las campañas de donaciones que se realizan por medio de alianzas con los partners son más efectivas que las recaudaciones por medio de redes sociales.
    - Una vez que Namá Conservation realiza el contacto para la donación, queda a la espera de que las perdona efectúe la donación.
2) ¿Como consolida la información de donaciones?
    - El personal de Namá destaca que el único medio para revisar los estatus o progresos de las campañas de donaciones es por medio de una solicitud hacia el contador o el tesorero de la organización ya que solo ellos tienen acceso a las cuentas.
3) ¿Cuáles son los medios para recaudar donaciones?

- Actualmente la ONG cuenta con dos medios para la recaudación de donaciones; el primero es por medio de transferencia bancaria hacia el banco nacional de Costa Rica y el segundo es por PayPal donde es su mayor fuente de ingresos debido a que reciben bastantes donaciones monetarias del extranjero.

4) ¿Como se consolida la información de las personas que quieren hacer voluntariados?
    - Namá Conservation carece de un medio de consolidación de información o una base de datos donde pueda almacenar la información de las personas quienes desean realizar voluntariados.
    - Al no tener un medio existente de consolidación para los voluntariados, muchas veces no realizan la coordinación del voluntariado con la persona por perdida de información del contacto.

4.4 BRECHAS O CONCLUSIONES DEL DIAGNOSTICO

A continuación, se detalla en la tabla 10 la situación actual, la brecha y la situación deseada.

Situación actual	Brecha	Situación deseada
No existe una página web donde Namá Conservation de a conocer la información principal de la organización.	Debe existir una página web con la información principal del conservatorio.	Tener una página web con la información principal de la ONG.
No existe un medio oficial de Namá Conservation para la presentación de las investigaciones científicas, proyectos, desarrollos o programas de voluntariados.	Se requiere realizar publicaciones en la página web para usarlo como el medio de comunicación oficial.	Tener un medio de comunicación oficial por el cual se pueda presentar los proyectos del conservatorio.

El proceso para la recaudación de las donaciones monetarias se realiza de forma pausada.	Se debe mejorar el proceso de las recaudaciones monetarias.	Reemplazar el proceso actual por una aplicación web que permita realizar la recaudación de las donaciones.
No existe una base de datos o un archivo centralizado de las personas que desean realizar voluntariados en los diferentes programas del conservatorio.	Se requiere de una base de datos o un medio centralizado de información para el almacenamiento de las personas que desean realizar voluntariados.	Tener un medio centralizado donde se pueda descargar la información de las personas que desean realizar voluntariado según los programas de la ONG.

Tabla 10 Brechas o conclusiones del diagnostico
Fuente: Elaboración propia

CAPÍTULO V: PROPUESTA DE PROYECTO

5.1 METODOLOGÍA ÁGIL

La metodología ágil implementada para el proyecto es la Extreme Programing, debido a que esta metodología está enfocada para los equipos pequeños de desarrollo, a comparación de Scrum que ocupa muchas figuras o roles presentes como el Scrum master, Product Owner o el equipo de desarrollo, quien cada uno de ellos tiene una función específica para el desarrollo del proyecto.

Por otro lado, la implementación de las metodologías tradicionales es un poco obsoleto en la actualidad, porque es muy difícil llevar una secuencia lineal del proyecto, requiere mucho tiempo para ver el trabajo finalizado y si se detecta algún error durante la etapa de pruebas puede que genere algún retroceso porque habría que realizar un rediseño y programación de los errores que fueron encontrados previamente.

Por lo tanto, Extreme Programming se acopla mejor al proyecto, porque esta metodología está orientada hacia las necesidades del cliente porque este, es implicado en el proceso dentro de cada iteración para validar o verificar que los avances tengan el resultado esperado, además que el proyecto es realizado solo por una persona.

5.1.1 Etapas de Extreme Programming

Exploración

Una vez que se han realizado las entrevistas para el proyecto y se hayan definido los requerimientos funcionales, requerimientos no funcionales, actores junto con toda la documentación necesaria, se inicia la etapa de exploración.

En la etapa de exploración se extrae el análisis de la documentación para realizar los casos de usos para determinar cuál es el funcionamiento correcto que debe tener la aplicación web para que pueda eficientizar los procesos actuales de la ONG Namá Conservation. Los casos de uso extraídos de la documentación son los siguientes:

- Módulo de donaciones
- Módulo de voluntariado

- Módulo de publicaciones
- Gestión de programas
- Gestión de intereses
- Gestión de usuarios

Además de la definición y la construcción de los casos de uso, se define la arquitectura con la que va a contar el proyecto, la arquitectura se define en los siguientes puntos:

Lenguaje de programación: El lenguaje de programación para las páginas web es HTML versión 5 con PHP, este lenguaje es complementado con JavaScript para poder dar funcionalidad a las páginas web y con CSS para darle diseño adaptable a cualquier tipo de dispositivo electrónico que tenga algún navegador web.

Framework: A pesar de que existen varios Frameworks para el desarrollo web como Asp.NET, Ruby on Rails o Symfony, el framework por defecto para el proyecto es Laravel en la versión 8.0, esta decisión fue técnicamente aprobada por Namá Conservation debido a que ellos cuentan con un partner comercial quienes le brindarán soporte a la aplicación con este framework una vez implementada la primera versión del proyecto. Además, que Laravel es beneficioso debido a su sistema de Middleware que permite la definición o autenticación de usuarios para el acceso de ciertas páginas, dominios y hasta los tipos de peticiones que se realizan al servidor, logrando aportar una serie de medidas de seguridad para evitar la filtración, divulgación o ingreso de la información no deseada.

Base de datos: La base de datos que son compatibles con Laravel son MYSQL, PostgreSQL o SQL Server, para el caso del proyecto, la base de datos a utilizar es MYSQL, debido a que Namá Conservation ya cuenta con un servicio de hosting, quien ofrece el motor de base de datos con MYSQL.

Editor de código: Para el desarrollo de la aplicación web se define el editor de código Visual Studio Code, porque este IDE tiene una licencia de software libre permisiva y que además es compatible con el lenguaje de programación definido para el proyecto.

Planificación

En la etapa de planificación se asigna la cantidad de iteraciones y el orden en el que se va a plasmar cada caso de uso en la aplicación web hasta llegar a la primera versión del sistema. Las iteraciones para realizar el proyecto son las siguientes:

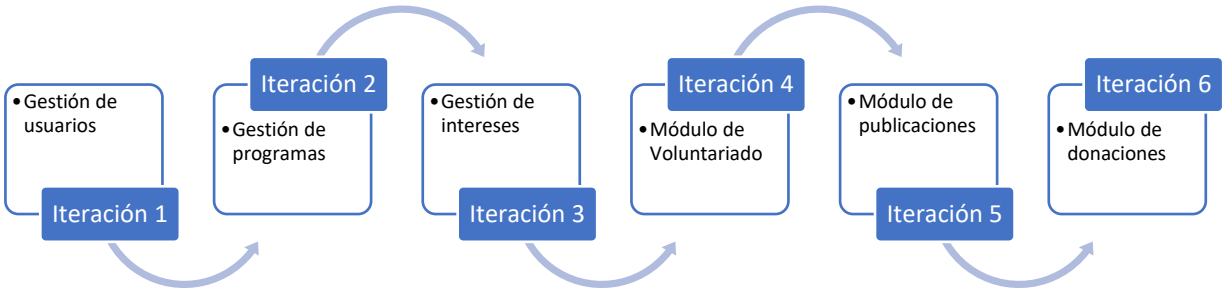


Figura 19 Diagrama de iteraciones
Fuente: Elaboración propia

Iteraciones

En la etapa de iteraciones se analiza un caso de uso en específico y sucesivamente se inicia con la codificación para poder plasmar la necesidad de cada caso de uso. No se avanza a la siguiente iteración hasta no completar el 100% de la funcionalidad del caso de uso. Una vez completada la programación del caso de uso, se realiza una reunión con el personal de Namá Conservation para presentar los avances del proyecto y así continuar con la siguiente iteración. Ver figura 20.

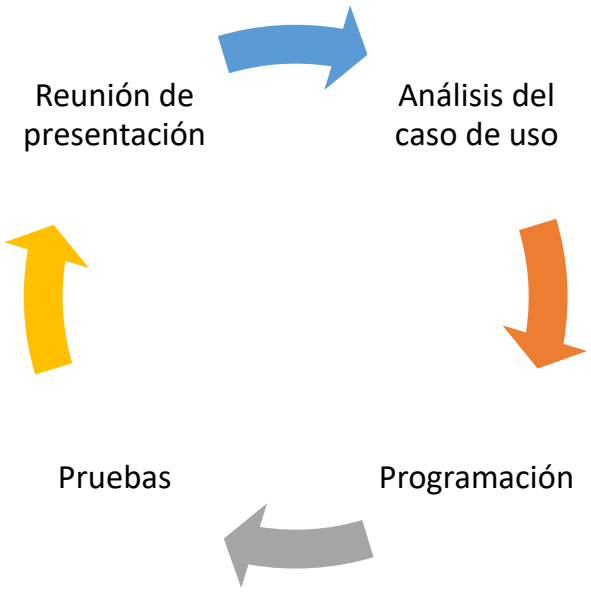


Figura 20 Proceso de iteración
Fuente: Elaboración propia

En caso de existir una solicitud de cambio tras la reunión con la ONG, se vuelve a reanalizar los casos de uso y se vuelve a completar el ciclo.

Implementación

Una vez terminadas todas las iteraciones, se realiza una presentación al personal de Nama Conservation donde se realizan pruebas de aceptación, si estas son aprobadas, se puede realizar la carga del proyecto al servidor y de esta forma entregar la primera versión de la aplicación web.

5.2 REQUERIMIENTOS

5.2.1 Identificación de actores

Los actores definidos para el uso y responsabilidades de la aplicación web se detalla a continuación:

ACT-01	Investigador Científico
Descripción	Funcionario encargado de realizar publicaciones o noticias del conservatorio, llevar seguimiento de las campañas de donaciones y reclutador de personas para los programas de voluntariados

Tabla 11 Actor 1
Fuente: Elaboración propia

ACT-02	Internauta
Descripción	Persona quien navega por internet, la misma puede realizar donaciones e inscripciones para los programas de voluntariados de Namá Conservation

Tabla 12 Actor 2
Fuente: Elaboración propia

5.2.2 Objetivos de funcionamiento

OBJ-01	Recaudación de donaciones
Descripción	El sistema debe realizar donaciones monetarias por vía PayPal y registrarlas para mostrar los avances a los investigadores científicos.
Importancia	Alta

Tabla 13 Objetivo de funcionamiento 1
Fuente: Elaboración propia

OBJ-02	Registro de Voluntariado
Descripción	El sistema debe realizar el registro de las personas que desean realizar voluntariados y estas se pueden visualizar para que los investigadores científicos puedan reclutarlas.
Importancia	Alta

Tabla 14 Objetivo de funcionamiento 2
Fuente: Elaboración propia

OBJ-03	Creación de publicaciones
Descripción	El sistema debe crear publicaciones donde pueda cargar imágenes, videos vía YouTube y noticias para mostrarlas en la página web.
Importancia	Alta

Tabla 15 Objetivo de funcionamiento 3
Fuente: Elaboración propia

OBJ-04	Creación de página web
Descripción	El sistema debe estar ligado a una página web, quien va a ser el medio de recolección de información.
Importancia	Alta

Tabla 16 Objetivo de funcionamiento 4
Fuente: Elaboración propia

5.2.3 Requerimientos funcionales

Requerimiento	REQF-001
Titulo	Integración con PayPal
Actores	ACT-02
Descripción	El sistema debe tener una integración con el servicio de PayPal para poder recibir donaciones, donde permita al usuario realizar la transacción vía pago con tarjeta de débito o crédito o desde la misma cuenta de PayPal.
Objetivos Asociados	OBJ-01, OBJ-04

Tabla 17 Requerimiento funcional 1
Fuente: Elaboración propia

Requerimiento	REQF-002
Titulo	Almacenamiento de donaciones
Actores	ACT-01, ACT-02
Descripción	El sistema debe almacenar la donación con la siguiente información: 1. Numero de referencia: Corresponde al identificador de la transferencia de PayPal. 2. Descripción: Nombre de la persona, organización o referencia del donante. 3. Comentario: Espacio para que el usuario pueda dejar algún comentario (Este espacio es opcional para el internauta) 4. Monto Donado: Cantidad de monto donado. 5. Estado: El estado de la transacción según PayPal. 6. Fecha: Fecha que se realizó la transacción
Objetivos Asociados	OBJ-01

Tabla 18 Requerimiento funcional 2
Fuente: Elaboración propia

Requerimiento	REQF-003
Titulo	Visualización de donaciones
Actores	ACT-01
Descripción	El sistema debe mostrar las donaciones realizadas y las mismas pueden ser buscadas o filtradas según los campos almacenados.
Objetivos Asociados	OBJ-01

Tabla 19 Requerimiento funcional 3
Fuente: Elaboración propia

Requerimiento	REQF-004
Titulo	Reporte de donaciones
Actores	ACT-01
Descripción	El sistema debe de realizar un exportable de Excel de las donaciones realizadas según las fechas de inicio y fin indicadas. Las columnas deben contener la siguiente información: 1. Numero de referencia de PayPal. 2. Descripción. 3. Monto. 4. Estado de la donación. 5. Fecha con hora.
Objetivos Asociados	OBJ-01

Tabla 20 Requerimiento funcional 4
Fuente: Elaboración propia

Requerimiento	REQF-005
Titulo	Almacenamiento de personas para programas de voluntariados
Actores	ACT-02
Descripción	El sistema debe almacenar la información de las personas que desean realizar voluntariados con Namá Conservation. La información que deben ingresar es la siguiente: 1. Nombre: Nombre de la persona. 2. Apellidos: Apellidos de la persona. 3. Nacionalidad: Nacionalidad de la persona. 4. Correo: Correo de la persona. 5. País: País de residencia. 6. Ciudad: Ciudad de residencia. 7. Dirección: Dirección de residencia. 8. Fecha de nacimiento. 9. Expectativas: Espacio para que el internauta explique las expectativas que tiene al realizar un programa de voluntariado. 10. Habilidades: Espacio para que el internauta describa las habilidades que pueda aportar a la organización. 11. Intereses: Selección de los intereses que puede realizar dentro de la organización.

	12. Programas: Selección de los programas que tiene Namá Conservation.
Objetivos Asociados	OBJ-02, OBJ-04

Tabla 21 Requerimiento funcional 5
Fuente: Elaboración propia

Requerimiento	REQF-006
Titulo	Visualización de voluntariados
Actores	ACT-01
Descripción	El sistema debe mostrar las personas que desean realizar algún programa de voluntariado. La misma información puede ser buscada o filtrada según los campos almacenados.
Objetivos Asociados	OBJ-02

Tabla 22 Requerimiento funcional 6
Fuente: Elaboración propia

Requerimiento	REQF-007
Titulo	Reporte de voluntariado
Actores	ACT-01
Descripción	El sistema debe de realizar un exportable de Excel con las personas que quieren realizar voluntariado. El Excel descargado debe ser filtrado según las fechas de inicio y fin en el que la persona ingreso la solicitud. Las columnas deben contener la siguiente información: 1. Nombre de la persona. 2. Apellido. 3. Correo. 4. Nacionalidad. 5. País. 6. Ciudad. 7. Dirección. 8. Fecha de nacimiento. 9. Expectativas. 10. Habilidades. 11. Intereses. 12. Programas seleccionados. 13. Fecha de solicitud.
Objetivos Asociados	OBJ-02

Tabla 23 Requerimiento funcional 7
Fuente: Elaboración propia

Requerimiento	REQF-008
Titulo	Eliminación de personas voluntarias
Actores	ACT-01
Descripción	El sistema debe poder eliminar las personas que desean realizar algún programa de voluntariado.
Objetivos Asociados	OBJ-02

Tabla 24 Requerimiento funcional 8
Fuente: Elaboración propia

Requerimiento	REQF-009
Titulo	Creación de publicaciones
Actores	ACT-01
Descripción	El sistema debe crear y almacenar publicaciones donde contenga la siguiente información: 1. Titulo. 2. Cuerpo del mensaje. 3. URL para el video. 4. Usuario quien pública. 5. Imágenes de la publicación.
Objetivos Asociados	OBJ-03, 0BJ-04

Tabla 25 Requerimiento funcional 9
Fuente: Elaboración propia

Requerimiento	REQF-010
Titulo	Modificación de publicaciones
Actores	ACT-01
Descripción	El sistema debe poder modificar la información de cualquier publicación.
Objetivos Asociados	OBJ-03, OBJ-04

Tabla 26 Requerimiento funcional 10
Fuente: Elaboración propia

Requerimiento	REQF-011
Titulo	Eliminación de publicaciones
Actores	ACT-01
Descripción	El sistema debe poder eliminar la información de cualquier publicación.
Objetivos Asociados	OBJ-03, OBJ-04

Tabla 27 Requerimiento funcional 11
Fuente: Elaboración propia

Requerimiento	REQF-012
Titulo	Visualización de publicaciones
Actores	ACT-01, ACT-02
Descripción	El sistema debe poder mostrar la información de cualquier publicación dentro de la página web.
Objetivos Asociados	OBJ-03, OBJ-04

Tabla 28 Requerimiento funcional 12
Fuente: Elaboración propia

Requerimiento	REQF-013
Titulo	Gestión de usuarios
Actores	ACT-01
Descripción	El sistema debe poder realizar la gestión de crear, eliminar, modificar y consultar la información de los usuarios. Lo datos requeridos son los siguientes: 1. Nombre del usuario. 2. Apellido del usuario. 3. Correo del usuario. 4. Contraseña. En caso de que el usuario olvide la contraseña, el sistema debe de enviar un correo de restablecimiento de contraseña.
Objetivos Asociados	OBJ-01, OBJ-02, OBJ-03, OBJ-04

Tabla 29 Requerimiento funcional 13
Fuente: Elaboración propia

Requerimiento	REQF-014
Titulo	Gestión de intereses
Actores	ACT-01
Descripción	El sistema debe poder realizar la gestión de crear, eliminar, modificar y consultar los intereses que tienen las personas para los programas de voluntariados. El espacio requerido es el nombre del interés.
Objetivos Asociados	OBJ-02

Tabla 30 Requerimiento funcional 14
Fuente: Elaboración propia

Requerimiento	REQF-015
Titulo	Gestión de programas
Actores	ACT-01
Descripción	El sistema debe poder realizar la gestión de crear, eliminar, modificar y consultar los programas que están disponibles en Namá Conservation. El espacio requerido es el nombre del programa.
Objetivos Asociados	OBJ-02

Tabla 31 Requerimiento funcional 15
Fuente: Elaboración propia

5.2.4 Requerimientos no funcionales

Requerimiento	REQNF-01
Titulo	Aplicación multinavegador
Descripción	La aplicación web debe funcionar correctamente en los navegadores existentes. Los principales son: • Google Chrome. • Internet Explorer. • Microsoft Edge. • Firefox. • Safari.

Tabla 32 Requerimiento no funcional 1
Fuente: Elaboración propia

Requerimiento	REQNF-02
Titulo	Libro de Marca
Descripción	La página web que está disponible al público debe tomar en cuenta el libro de marca de Namá Conservation.

Tabla 33 Requerimiento no funcional 2
Fuente: Elaboración propia

Requerimiento	REQNF-03
Titulo	Base de datos predeterminada
Descripción	La base de datos con la que el proyecto debe funcionar es MYSQL versión 5.7.24 o superior

Tabla 34 Requerimiento no funcional 3
Fuente: Elaboración propia

Requerimiento	REQNF-04
Titulo	Framework predeterminado
Descripción	El framework debe ser laravel versión 7 o superior.

Tabla 35 Requerimiento no funcional 4
Fuente: Elaboración propia

Requerimiento	REQNF-05
Titulo	Idioma predeterminado
Descripción	La aplicación web debe tener configurado el idioma ingles por defecto. La página web debe tener la opción para traducir a los idiomas: • Alemán • Checo • Chino • Español • Frances • Italiano • Ruso

Tabla 36 Requerimiento no funcional 5
Fuente: Elaboración propia

5.3 CASOS DE USO

5.3.1 Módulo de donaciones

Caso de uso	Módulo de donaciones	
Actores	ACT-01, ACT-02	
Descripción	El actor ACT-02 puede efectuar donaciones sin necesidad de realizar autenticación en la aplicación de Namá Conservation, este puede ser una persona, organización o un anónimo. Por otro lado, el actor ACT-01 puede consultar, eliminar, filtrar y descargar la información de las donaciones.	
Requerimientos asociados	REQF-001, REQF-002, REQF-003, REQF-004	
Precondición	El actor ACT-01 requiere de iniciar sesión previamente en el sistema. El actor ACT-02 no requiere del inicio de sesión en el sistema.	
Post condición	La información es registrada, sucesivamente visualizada y descargada.	
Caso normal	**Secuencia**	**Acción**
	1	El actor ACT-02 ingresa a la página principal de Namá Conservation y oprime la opción de donación.
	2	El sistema solicita y valida la información al actor ACT-02 para realizar la donación.

	3	El actor ACT-02 elige si la donación es por pago con tarjeta de crédito/débito o si es por medio de los fondos disponibles de PayPal.
	4	El actor ACT-02 realiza la donación y el sistema registra la donación.
	5	El actor ACT-01 puede visualizar la información de la donación.
	6	El actor 01 puede descargar la información en formato de Excel.
	7	El actor ACT-01 puede eliminar la información en caso de una devolución.

Tabla 37 Caso de uso del módulo de donaciones
Fuente: Elaboración propia

5.3.2 Módulo de voluntariado

Caso de uso	Módulo de voluntariado	
Actores	ACT-01, ACT-02	
Descripción	El actor ACT-02 puede registrarse para los programas de voluntariados. Por otro lado, el actor ACT-01 puede consultar, eliminar, filtrar y descargar la información de las personas que se han registrado para realizar voluntariados.	
Requerimientos asociados	REQF-005, REQF-006, REFQ-007, REFQ-008	
Precondición	El actor ACT-01 requiere de iniciar sesión previamente en el sistema. El actor ACT-02 no requiere del inicio de sesión en el sistema.	
Post condición	La información es registrada, sucesivamente visualizada, descargada y eliminada.	
Caso normal	**Secuencia**	**Acción**
	1	El actor ACT-02 ingresa a la página principal de Namá Conservation y selecciona la opción de voluntariado.
	2	El sistema le solicita al actor ACT-02 la información necesaria para inscribirse como voluntario.
	3	El sistema verifica y almacena la información que registro el actor ACT-02.

	4	El actor ACT-01 puede visualizar la información ingresada del ACT-02.
	5	El actor 01 puede descargar la información en formato de Excel.
	6	El actor ACT-01 puede eliminar la información en caso de que el ACT-02 sea rechazado.

Tabla 38 Caso de uso del módulo de voluntariado
Fuente: Elaboración propia

5.3.3 Módulo de publicaciones

Caso de uso	Módulo de publicaciones	
Actores	ACT-01, ACT-02	
Descripción	El actor ACT-01 puede crear, consultar, modificar y eliminar cualquier publicación. Por otro lado, el actor ACT-02 solamente puede visualizar.	
Requerimientos asociados	REQF-009, REQF-010, REFQ-011, REFQ-012	
Precondición	El actor ACT-01 requiere de iniciar sesión previamente en el sistema. El actor ACT-02 no requiere del inicio de sesión en el sistema.	
Post condición	La información creada puede ser modificada, eliminada y visualizada.	
Caso normal	**Secuencia**	**Acción**
	1	El actor ACT-01 ingresa al sistema y selecciona la opción de publicaciones.
	2	El actor ACT-01 selecciona la opción de crear publicación.
	3	El sistema solicita los campos requeridos para realizar la publicación.
	4	El sistema almacena la información luego de que el actor ACT-01 seleccione la opción de guardar.

	5	El actor ACT-01 puede ver la lista de las publicaciones creadas.
	6	El actor ACT-01 puede seleccionar una publicación para visualizar el contenido y poder modificarla.
	7	El sistema almacena los cambios realizados en la publicación.
	8	El actor ACT-01 puede eliminar la publicación y el sistema solicita confirmación antes de realizar la eliminación.
	9	El actor ACT-02 puede visualizar todas las publicaciones que no han sido eliminadas.

Tabla 39 Caso de uso del módulo de publicaciones
Fuente: Elaboración propia

5.3.4 Gestión de programas

Caso de uso	Gestión de programas	
Actores	ACT-01, ACT-02	
Descripción	El actor ACT-01 puede crear, consultar, modificar y eliminar cualquier tipo de programa. Por otro lado, el actor ACT-02 puede seleccionar los diferentes programas en los que desea participar cuando se registra en un voluntariado.	
Requerimientos asociados	REQF-005, REQF-015	
Precondición	El actor ACT-01 requiere de iniciar sesión previamente en el sistema. El actor ACT-02 no requiere del inicio de sesión en el sistema.	
Post condición	La información creada puede ser modificada, eliminada por actor ACT-01 y ser seleccionada por el actor ACT-02.	
Caso normal	**Secuencia**	**Acción**
	1	El actor ACT-01 ingresa al sistema y selecciona la opción de programas.
	2	El actor ACT-01 selecciona la opción de crear programa.
	3	El sistema solicita los campos requeridos para crear el nuevo programa.
	4	El sistema almacena la información luego de que el actor ACT-01 seleccione la opción de guardar.

	5	El actor ACT-01 puede ver la lista de los programas disponibles.
	6	El actor ACT-01 puede seleccionar un programa para modificar el contenido.
	7	El sistema almacena los cambios realizados en el programa.
	8	El actor ACT-01 puede eliminar el programa en caso de ser necesario.
	9	El actor ACT-02 puede seleccionar los programas no eliminados a la hora de registrarse como voluntario.

Tabla 40 Caso de uso de la gestión de programas
Fuente: Elaboración propia

5.3.5 Gestión de intereses

Caso de uso	Gestión de intereses	
Actores	ACT-01	
Descripción	El actor ACT-01 puede crear, consultar, modificar y eliminar cualquier tipo de interés para realizar voluntariado. Por otro lado, el actor ACT-02 puede seleccionar los diferentes intereses en los que desea realizar cuando se registra en un voluntariado.	
Requerimientos asociados	REQF-005, REQF-014	
Precondición	El actor ACT-01 requiere de iniciar sesión previamente en el sistema. El actor ACT-02 no requiere del inicio de sesión en el sistema.	
Post condición	La información creada puede ser modificada, eliminada por actor ACT-01 y ser seleccionada por el actor ACT-02.	
Caso normal	**Secuencia**	**Acción**
	1	El actor ACT-01 ingresa al sistema y selecciona la opción de intereses.
	2	El actor ACT-01 selecciona la opción de crear interés.
	3	El sistema solicita los campos requeridos para crear el nuevo interés.

	4	El sistema almacena la información luego de que el actor ACT-01 seleccione la opción de guardar.
	5	El actor ACT-01 puede ver la lista de los intereses disponibles.
	6	El actor ACT-01 puede seleccionar un interés para modificar el contenido.
	7	El sistema almacena los cambios realizados en el interés luego de que el actor ACT-01 oprime la opción de guardar.
	8	El actor ACT-01 puede eliminar el interés en caso de ser necesario.
	9	El actor ACT-02 puede seleccionar los intereses no eliminados a la hora de registrarse como voluntario.

Tabla 41 Caso de uso de la gestión de intereses
Fuente: Elaboración propia

5.3.6 Gestión de usuarios

Caso de uso	Gestión de usuarios	
Actores	ACT-01	
Descripción	El actor ACT-01 puede crear, modificar e inactivar a los usuarios del sistema.	
Requerimientos asociados	REQF-013	
Precondición	El actor ACT-01 requiere de iniciar sesión previamente en el sistema. En caso de olvido de contraseña, el sistema debe de enviar un correo de restablecimiento si existe el correo en la base de datos.	
Post condición	Los usuarios creados pueden ser modificados, o inactivados.	
Caso normal	**Secuencia**	**Acción**
	1	El actor ACT-01 ingresa al sistema y selecciona la opción de usuarios.
	2	El actor ACT-01 selecciona la opción de crear usuario nuevo.
	3	El sistema solicita los campos requeridos para crear al nuevo usuario.
	4	El sistema almacena la información luego de que el actor ACT-01 seleccione la opción de guardar.
	5	El actor ACT-01 puede ver la lista de los usuarios.

	6	El actor ACT-01 puede seleccionar un usuario para modificar el contenido.
	7	El sistema almacena los cambios realizados en el usuario luego de que el actor ACT-01 oprime la opción de guardar.
	8	El actor ACT-01 puede inactivar el usuario en caso de ser necesario.
	9	Si el usuario es inactivado, el sistema no debe permitir el acceso a la creación de publicaciones, reportes de donaciones, voluntariados u otras opciones del sistema.

Tabla 42 Caso de uso de la gestión de usuarios
Fuente: Elaboración propia

5.4 DIAGRAMAS DE CASOS DE USO

5.4.1 Módulo de donaciones

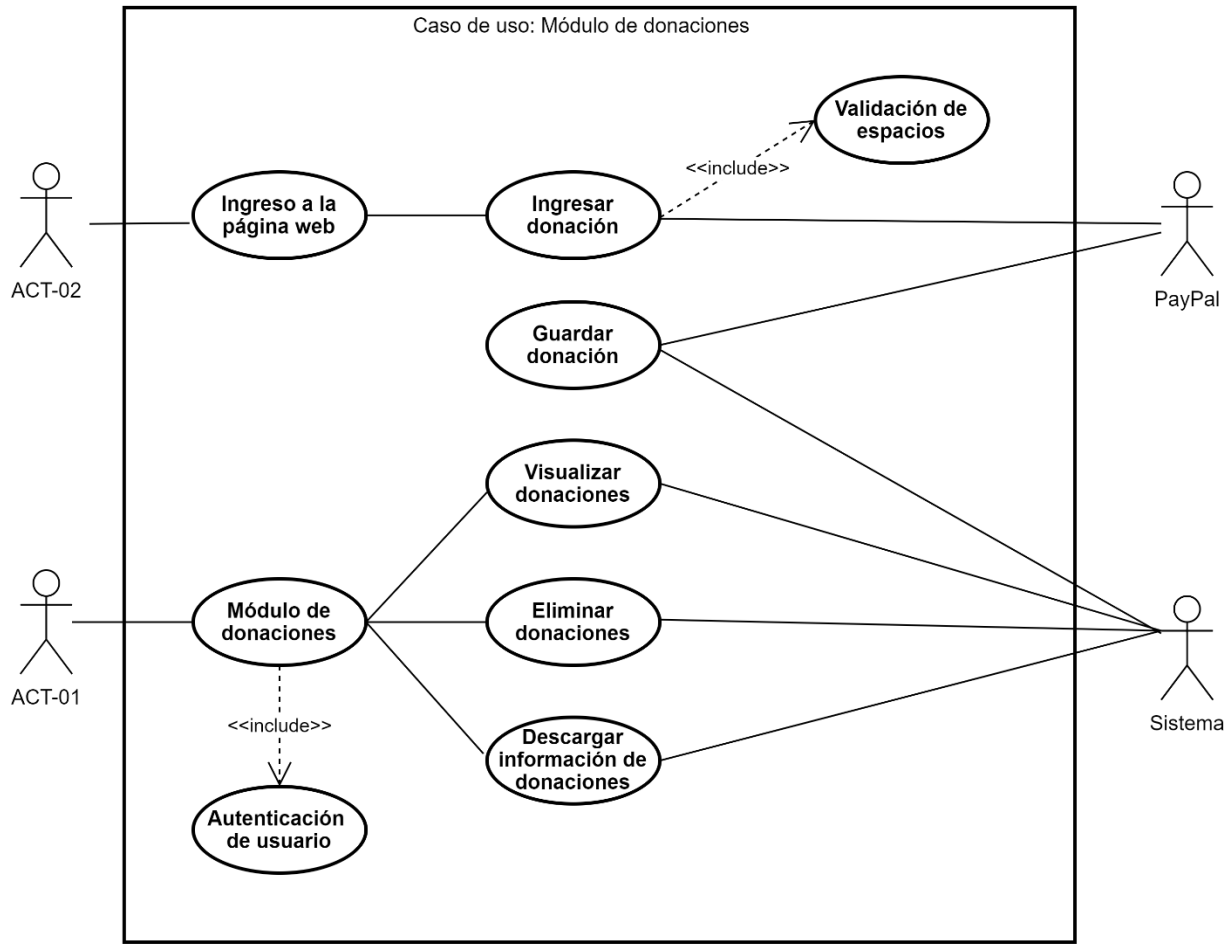


Figura 21 Diagrama de caso de uso de donaciones
Fuente: Elaboración propia

5.4.2 Módulo de voluntariado

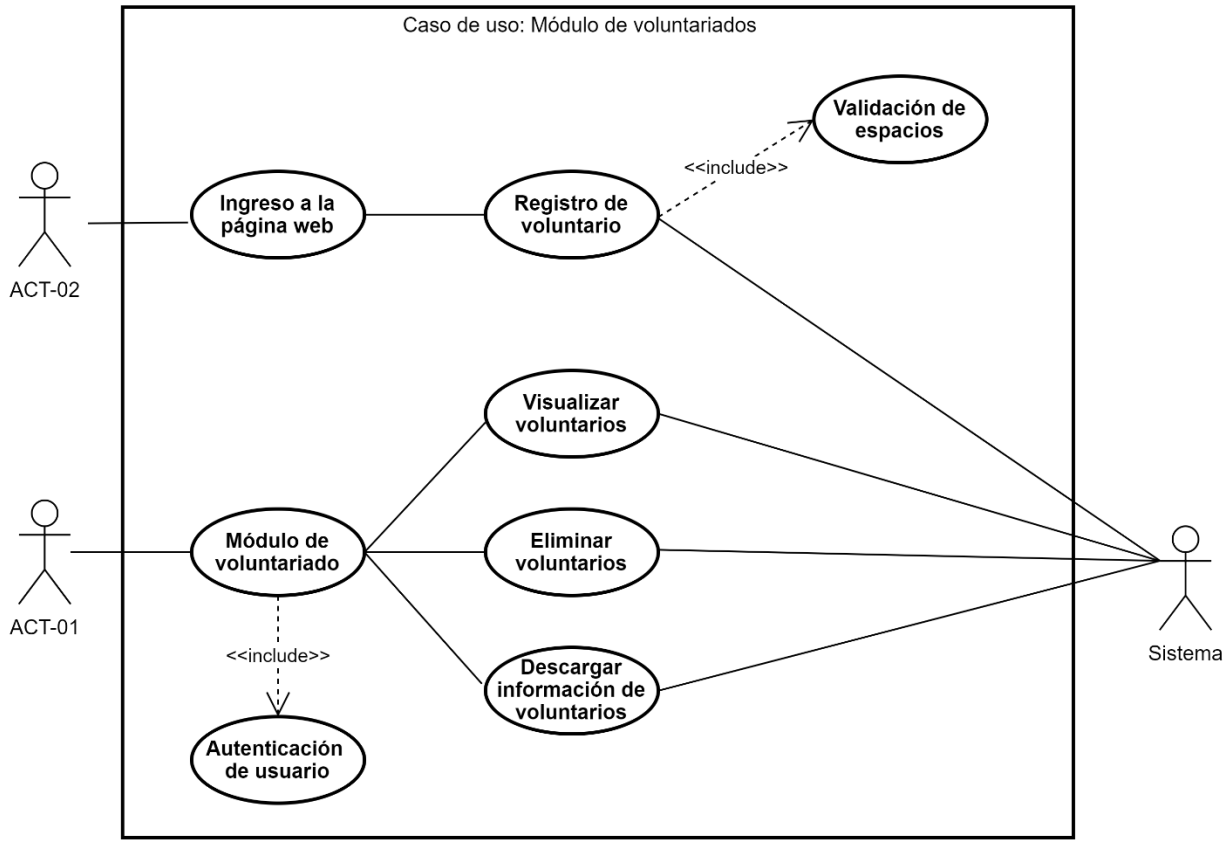


Figura 22 Diagrama de caso de uso de voluntariado
Fuente: Elaboración propia

5.4.3 Módulo de publicaciones

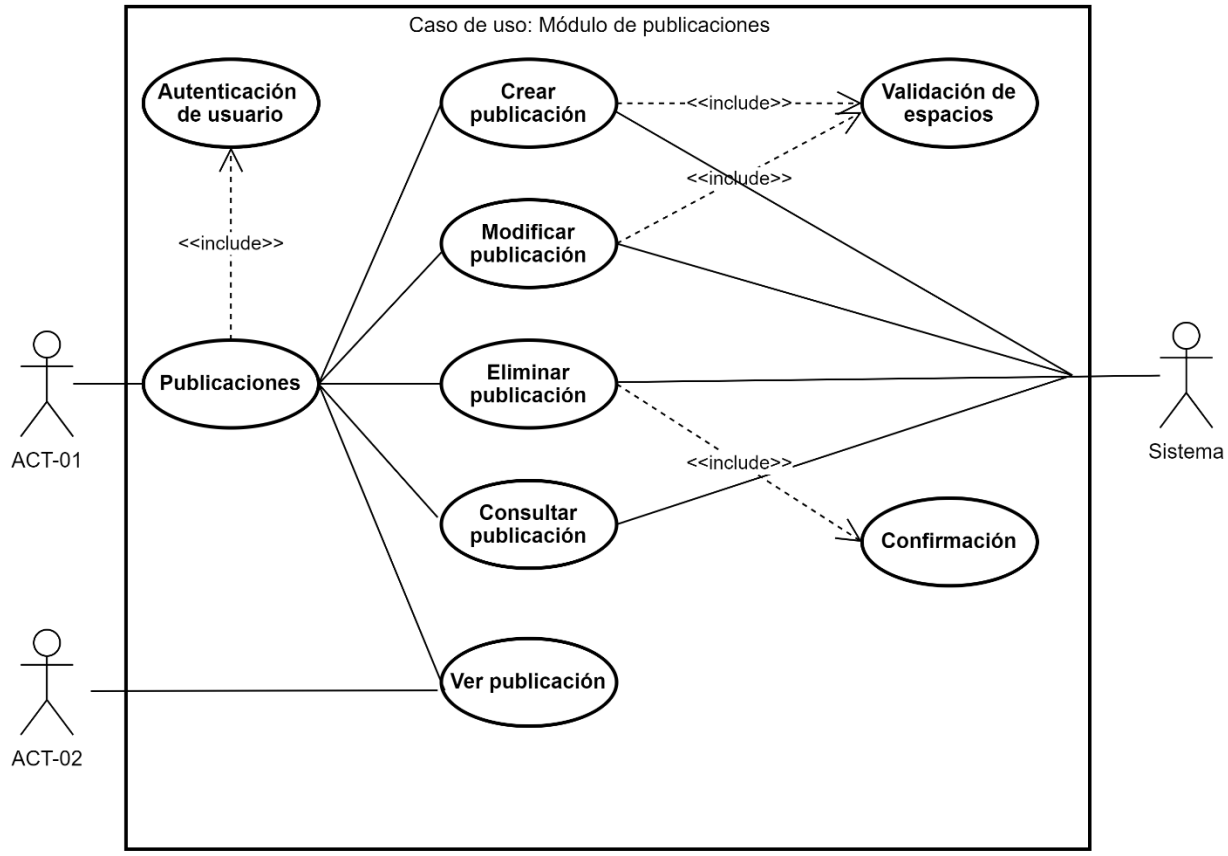


Figura 23 Diagrama de caso de uso de publicaciones
Fuente: Elaboración propia

5.4.4 Gestión de programas

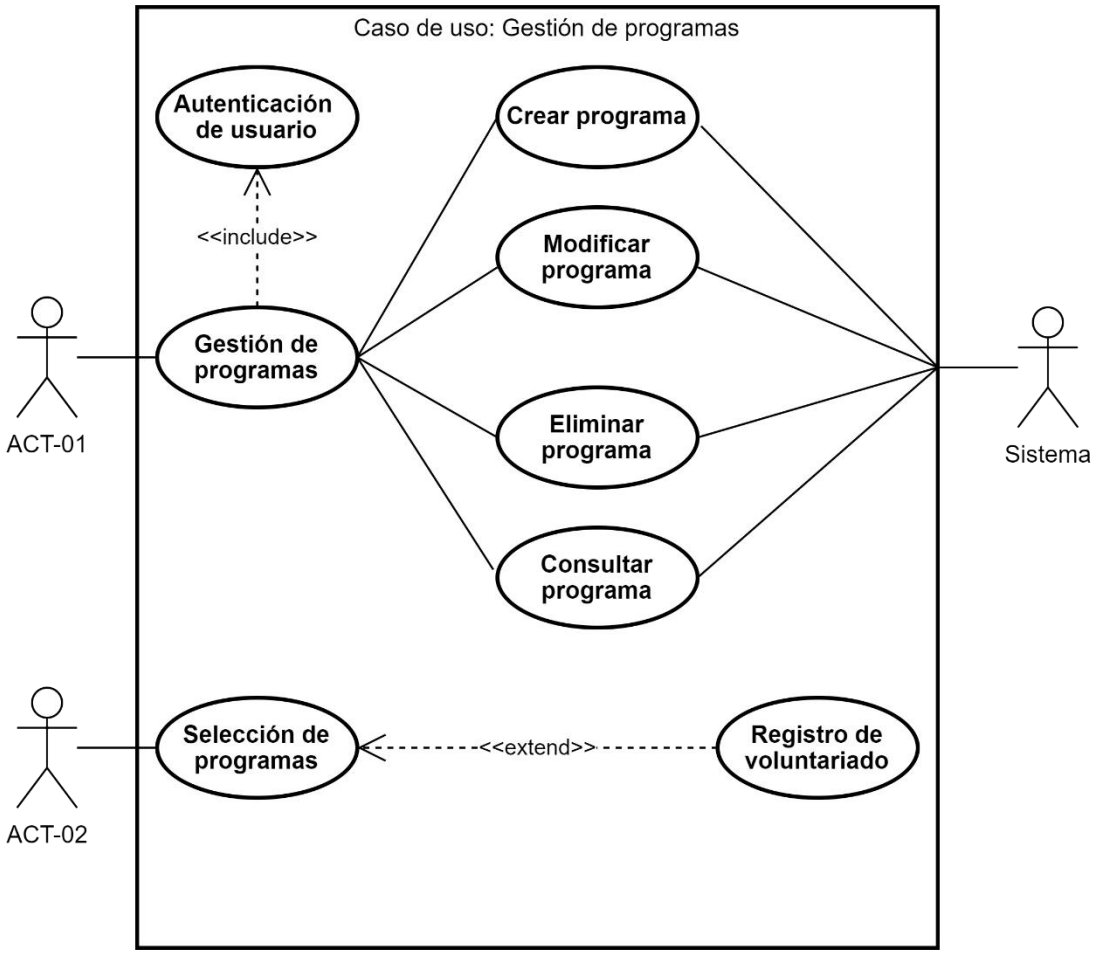


Figura 24 Diagrama de caso de uso gestión de programas
Fuente: Elaboración propia

5.4.5 Gestión de intereses

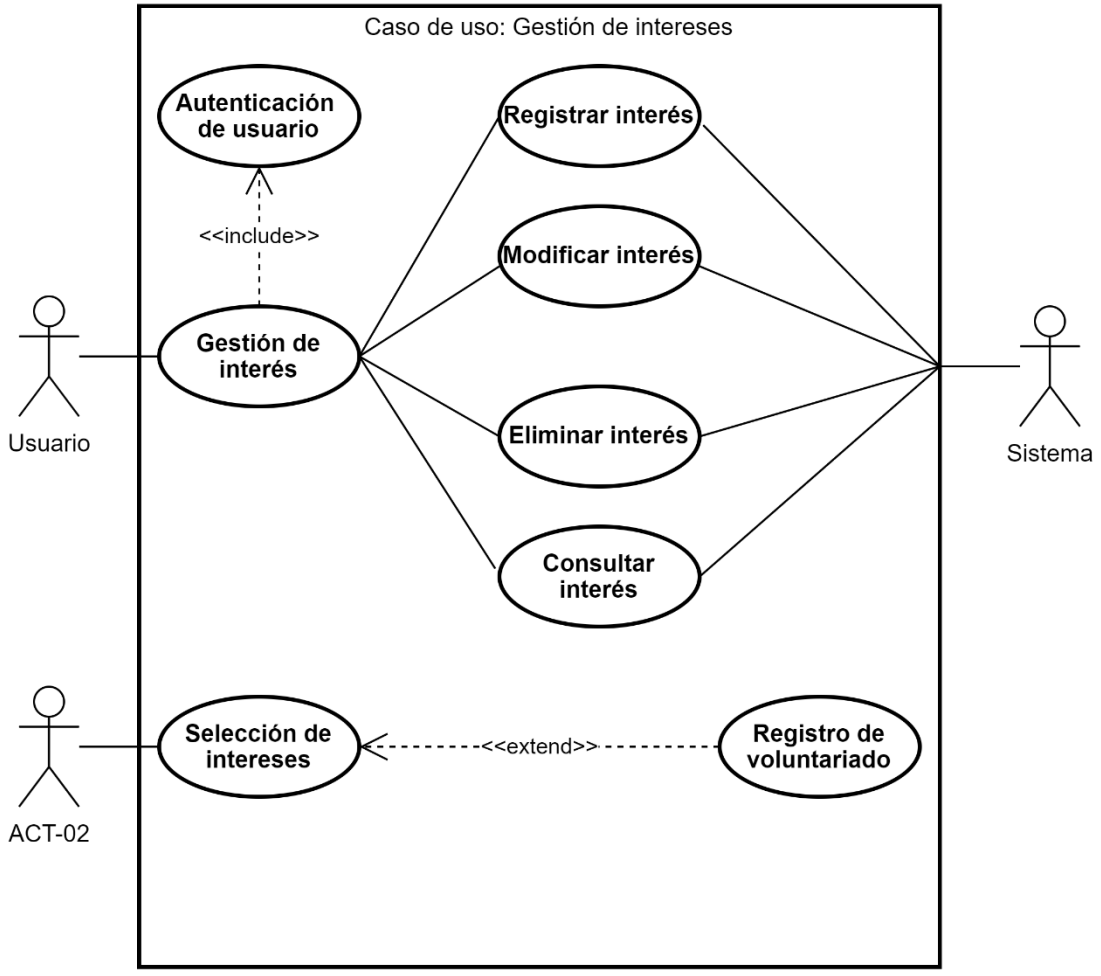


Figura 25 Diagrama de caso de uso gestión de intereses
Fuente: Elaboración propia

5.4.6 Gestión de usuarios

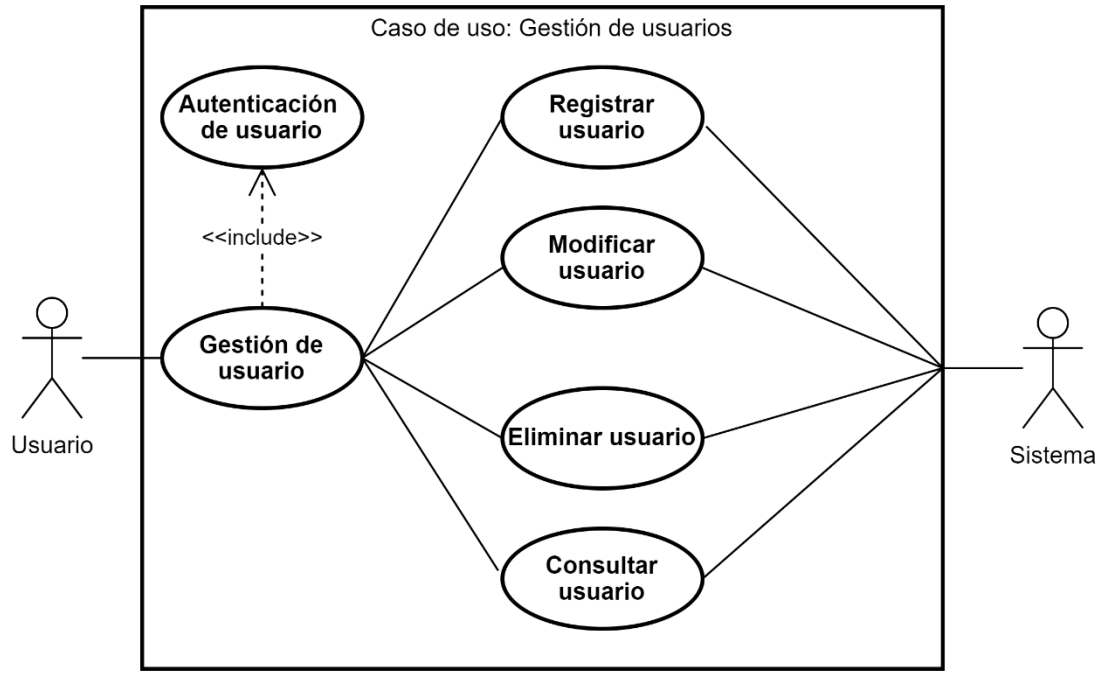


Figura 26 Diagrama de caso de uso gestión de usuarios
Fuente: Elaboración propia

5.5 DISEÑO

5.5.1 Diseño de base de datos

La base de datos fue realizada en MYSQL donde la ONG no proporciono ningún estándar para el diseño de la base de datos, solamente que estuviera en el idioma Ingles por defecto.

5.5.2 Diagrama de base de datos entidad relación

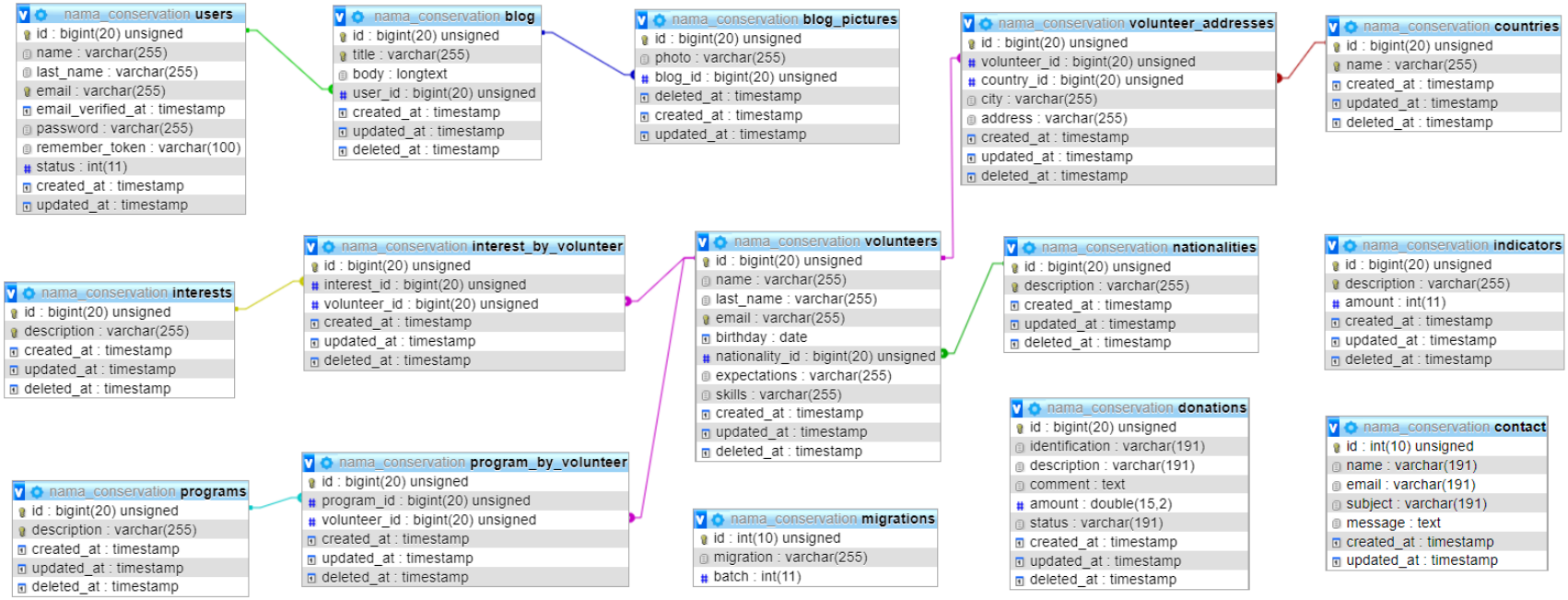


Figura 27 Diagrama de la base de datos entidad relación
Fuente: Elaboración propia

5.5.3 Diccionario de datos

Nombre de la tabla	users	
Descripción	Tabla de los usuarios que pueden acceder al sistema.	
Relaciones	blog	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador interno del usuario
name	varchar(255)	Nombre del usuario
last_name	varchar(255)	Apellidos del usuario
email	varchar(255)	Correo del usuario
email_verified_at	timestamp	Fecha de verificación del correo
password	varchar(255)	Contraseña del usuario
remember_token	varchar(100)	Llave para almacenar el inicio de sesión
status	int(11)	Estatus del usuario
created_at	timestamp	Fecha en la que fue creado el usuario
updated_at	timestamp	Fecha en la que fue actualizado el usuario

Tabla 43 Diccionario de datos: tabla de users
Fuente: Elaboración propia

Nombre de la tabla	blog	
Descripción	Tabla para las publicaciones del conservatorio	
Relaciones	blog_pictures	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador de la publicación
title	varchar(255)	Título de la publicación
body	longtext	Cuerpo o mensaje de la publicación
user_id	bigint(20)	Usuario quien creo la publicación
created_at	timestamp	Fecha en la que fue creada la publicación
updated_at	timestamp	Fecha en la que fue modificada la publicación
deleted_at	timestamp	Fecha en la que fue eliminada la publicación

Tabla 44 Diccionario de datos: tabla de blog
Fuente: Elaboración propia

Nombre de la tabla	blog_pictures	
Descripción	Tabla para almacenar las diferentes fotografías de cada publicación	
Relaciones	Ninguna	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador de la fotografía
photo	varchar(255)	Nombre de la fotografía
blog_id	bigint(20)	Publicación a la que está relacionada la fotografía
deleted_at	timestamp	Fecha en la que fue eliminada la fotografía
created_at	timestamp	Fecha en la que fue creada la fotografía
updated_at	timestamp	Fecha en la que fue modificada la fotografía

Tabla 45 Diccionario de datos: tabla de blog_pictures
Fuente: Elaboración propia

Nombre de la tabla	volunteers	
Descripción	Tabla de almacenamiento de las personas que quieren realizar voluntariados	
Relaciones	nationalities	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador interno del sistema de la persona
name	varchar(255)	Nombre de la persona
last_name	varchar(255)	Apellido de la persona
email	varchar(255)	Correo de la persona
birthday	date	Nacimiento de la persona
nationality_idÍndice	bigint(20)	Nacionalidad a la que está relacionada la persona
expectations	varchar(255)	Expectativas de la persona
skills	varchar(255)	Habilidades de la persona
deleted_at	timestamp	Fecha de creación
created_at	timestamp	Fecha de modificación
updated_at	timestamp	Fecha de eliminación

Tabla 46 Diccionario de datos: tabla de volunteers
Fuente: Elaboración propia

Nombre de la tabla	volunteer_addresses	
Descripción	Tabla para almacenar las direcciones de residencia de las personas	
Relaciones	volunteers, countries	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador interno
volunteer_id	bigint(20)	Relación con la persona voluntaria
country_id	bigint(20)	Relación con el país de residencia de la persona
city	varchar(255)	Ciudad de la persona
address	varchar(255)	Dirección de la persona
created_at	timestamp	Fecha en la que fue creada la dirección de la persona
updated_at	timestamp	Fecha en la que fue modificada la dirección de la persona
deleted_at	timestamp	Fecha en la que fue eliminada la dirección de la persona

Tabla 47 Diccionario de datos: tabla de volunteer_addresses
Fuente: Elaboración propia

Nombre de la tabla	countries	
Descripción	Tabla que almacena los países	
Relación	Ninguna	
Nombre	Tipo	Descripción
Id	bigint(20)	Identificador del país
Name	varchar(255)	Nombre del país
created_at	timestamp	Fecha en la que fue creado el país
updated_at	timestamp	Fecha en la que fue modificado el país
deleted_at	timestamp	Fecha en la que fue eliminado el país

Tabla 48 Diccionario de datos: tabla de countries
Fuente: Elaboración propia

Nombre de la tabla	nationalities	
Descripción	Tabla que almacena las nacionalidades	
Relación	Ninguna	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador de la nacionalidad
description	varchar(255)	Nombre de la nacionalidad
created_at	timestamp	Fecha de creación
updated_at	timestamp	Fecha de modificación
deleted_at	timestamp	Fecha de eliminación

Tabla 49 Diccionario de datos: tabla de nationalities
Fuente: Elaboración propia

Nombre de la tabla	interest_by_volunteer	
Descripción	Tabla para almacenar los diferentes intereses de la persona por realizar voluntariado	
Relación	volunteers, interests	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador de cada interés del voluntario
interest_id	bigint(20)	Relación con el interés
volunteer_id	bigint(20)	Relación con el voluntario
created_at	timestamp	Fecha en la que fue creado
updated_at	timestamp	Fecha en la que fue modificado
deleted_at	timestamp	Fecha en la que fue eliminado

Tabla 50 Diccionario de datos: tabla de interest_by_volunteer
Fuente: Elaboración propia

Nombre de la tabla	interest	
Descripción	Tabla que almacena los diferentes intereses que pueden tener las personas a la hora de hacer voluntariados	
Relación	Ninguna	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador del interés
description	varchar(255)	Nombre o descripción del interés
created_at	timestamp	Fecha de creación del interés
updated_at	timestamp	Fecha de modificación del interés
deleted_at	timestamp	Fecha de eliminación del interés

Tabla 51 Diccionario de datos: tabla de interest
Fuente: Elaboración propia

Nombre de la tabla	program_by_volunteer	
Descripción	Tabla que almacena cada programa en el que está interesado la persona en realizar voluntariado	
Relación	volunteers, programs	
Nombre	Tipo	Descripción
Id	bigint(20)	Identificador de cada programa del voluntario
program_id	bigint(20)	Relación con el programa
volunteer_id	bigint(20)	Relación con el voluntario
created_at	timestamp	Fecha de creación
updated_at	timestamp	Fecha de modificación
deleted_at	timestamp	Fecha de eliminación

Tabla 52 Diccionario de datos: tabla de program_by_volunteer
Fuente: Elaboración propia

Nombre de la tabla	programs	
Descripción	Tabla que almacena los diferentes programas de voluntariados con los que cuenta Namá Conservation	
Relación	Ninguna	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador del programa
description	varchar(255)	Nombre o descripción del programa
created_at	timestamp	Fecha en la que fue creado el programa
updated_at	timestamp	Fecha en la que fue modificado el programa
deleted_at	timestamp	Fecha en la que fue eliminado el programa

Tabla 53 Diccionario de datos: tabla de programs
Fuente: Elaboración propia

Nombre de la tabla	donations	
Descripción	Tabla donde se almacena la información de las donaciones realizadas	
Relación	Ninguna	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador de la donación
identification	varchar(191)	Identificador o código de referencia de PayPal
description	varchar(191)	Nombre de la organización, persona o descripción de la donación
comment	text	Comentario acerca de los proyectos o programas de Namá Conservation
amount	double(15,2)	Monto de la donación
status	varchar(191)	Estado de PayPal de la donación
created_at	timestamp	Fecha de creación
updated_at	timestamp	Fecha de modificación
deleted_at	timestamp	Fecha de la eliminación

Tabla 54 Diccionario de datos: tabla de donations
Fuente: Elaboración propia

Nombre de la tabla	indicators	
Descripción	Tabla para los indicadores principales de la página web	
Relación	Ninguna	
Nombre	Tipo	Descripción
id	bigint(20)	Identificador del indicador
description	varchar(255)	Nombre o descripción del indicador
amount	int(11)	Monto del indicador
created_at	timestamp	Fecha de creación del indicador
updated_at	timestamp	Fecha de modificación del indicador
deleted_at	timestamp	Fecha de eliminación del indicador

Tabla 55 Diccionario de datos: tabla de indicators
Fuente: Elaboración propia

Nombre de la tabla	contact	
Descripción	Tabla para el almacenamiento de los correos enviados por medio de la página web	
Relación	Ninguna	
Nombre	Tipo	Descripción
id	int(10)	Identificador del correo
name	varchar(191)	Nombre de la persona que envía el correo
email	varchar(191)	Correo de la persona
subject	varchar(191)	Título del correo
message	text	Cuerpo o mensaje del coreo
created_at	timestamp	Fecha de creación del correo
updated_at	timestamp	Fecha de actualización del correo

Tabla 56 Diccionario de datos: tabla de contact
Fuente: Elaboración propia

Nombre de la tabla	migrations	
Descripción	Tabla por defecto de Laravel para las versiones de cambios de la base de datos	
Relación	Ninguna	
Nombre	Tipo	Descripción
Id	int(10)	Identificador de la migración
migration	varchar(255)	Nombre de la migración
batch	int(11)	El número del versionado de los cambios de la base de datos

Tabla 57 Diccionario de datos: tabla de migrations
Fuente: Elaboración propia

5.6 DESARROLLO

La aplicación web se desarrolló con el framework de Laravel en la versión 8.0 con PHP versión 73.1.1, utilizando el IDE Visual Studio Code como editor de código y con el motor de base de datos de MYSQL.

5.6.1 Conexión de base de datos

Para la conexión de la base de datos con el framework, Laravel realiza la comunicación por medio del archivo .ENV donde define el nombre de la base de datos de MYSQL, junto con el usuario de base de datos y la contraseña para el acceso de la información, como lo muestra la figura 28.

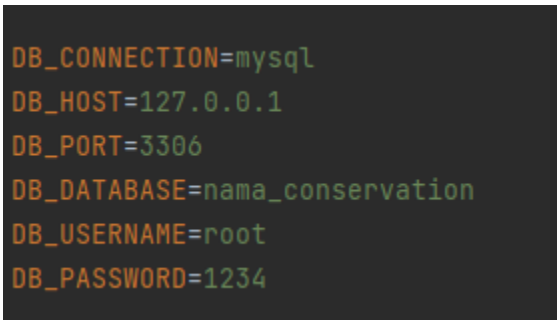


Figura 28 Conexión de Laravel con MYSQL
Fuente: Elaboración propia

Una vez definida la base de datos en el framework de Laravel. se asignan las tablas a cada modelo con el que va a estar relacionado, con excepción del modelo de usuarios porque este ya el framework lo tiene asignado por defecto.

5.6.2 Estructura del proyecto en Laravel

El proyecto creado con de Laravel contiene la siguiente estructura:

- App: En esta carpeta se almacenaron los controladores, modelos y clases para la creación de los reportes en Excel.
- Config: Carpeta por defecto de Laravel para diferentes configuraciones del sistema.
- Database: En esta carpeta se almacenan las migraciones de Laravel.
- Public: En esta carpeta se almacenaron todas las imágenes, JavaScript, CSS y librerías que correspondan a diseño de las páginas web.
- Resources: En esta carpeta se almacenaron las vistas de HTML a mostrar.
- Routes: Esta carpeta es por defecto de Laravel para realizar la conexión de las direcciones o URL con respecto a los controladores mediante las peticiones GET, POST, DELETE entre otros.

5.6.3 Librerías del proyecto

Para el diseño de las páginas web que van a estar disponibles para todos los cibernautas se utilizó la librería de Materialize para darle flexibilidad y adaptabilidad a las páginas web dentro de cualquier tipo de dispositivo de donde sea visitado el proyecto. Por otro lado, para las vistas que son únicas para los usuarios de Namá Conservation, se utilizó la librería de Bootstrap que es más versátil para la creación de formularios y funciones administrativas, además que es compatible con la librería Vue.js.

Para la funcionalidad de las páginas web se utilizó la librería de Vue.js porque tienen componentes que ayudan a realizar las tareas asignadas como consultas HTTP, componentes para cargar archivos al servidor y mensajes de dialogo que permiten darle una buena interacción con los usuarios.

Y por último para las creaciones de publicaciones se utilizó la librería de Summernote que tiene integración con jQuery, esta librería es fundamental para el proyecto porque permite que el editor de texto sea totalmente dinámico permitiendo a el usuario la selección de diferentes tipos de letra, tamaño, colores entre otras funcionalidades, convirtiendo el texto ingresado en código HTML para su respectivo almacenamiento y publicación.

5.6.4 Webservice de PayPal

Para que el proyecto pueda realizar las donaciones monetarias, se utilizó el Webservice de PayPal, donde la petición HTTP tipo Post por medio del URL https://www.paypal.com/cgi-bin/webscr solicita los datos necesarios para generar la donación para la ONG. Ver tabla 58.

Espacio solicitado	Descripción
business	Nombre de la cuenta de PayPal que normalmente es el correo con el que usa la empresa para el uso de PayPal.
cmd	En este espacio se colocó el tipo de transacción. Para el caso de las donaciones, PayPal solicita que el espacio por defecto sea _donations.
item_name	En este espacio se coloca el nombre de la persona, alias o nombre de la organización quien desea realizar la donación.
custom	El espacio de custom se colocó el comentario opcional que la persona quiera dejar por la donación.
currency_code	Este espacio corresponde al tipo de moneda a realizar la donación. El valor utilizado es USD que hace referencia a la cuenta en dólares.
amount	El monto en dólares que la persona desea realizar en la donación.
notify_url	En este espacio se coloca el URL de la aplicación de Namá Conservation donde quiere que sea notificado por la donación. Para el caso del proyecto no se contempló ninguna notificación.
cancel_return	En este espacio se envía la dirección URL de la aplicación de Namá Conservation por si el usuario decide cancelar la donación.
return	En este espacio va el URL de redireccionamiento si la donación se ejecutó correctamente.

Tabla 58 Webservice de PayPal
Fuente: Elaboración propia

Para la etapa de pruebas del proyecto se utilizó el ambiente de pruebas de PayPal Sand Box, para ello se cambia el URL tipo POST por https://www.sandbox.paypal.com/cgi-bin/webscr. Además, el ambiente de desarrollo de Sandbox permite la creación de cuentas PayPal y creación de tarjetas de crédito o débito para realizar pruebas.

Una vez enviada la información por medio del Webservice, PayPal se encarga de realizar la transacción de pago, ya sea por tarjeta de débito/crédito o por la misma cuenta de PayPal, al ser efectuada la donación, PayPal envía la información de la tabla 59 donde el controlador de las donaciones del proyecto recibe esta información y la almacena en la base de datos.

Espacio enviado	Descripción
tx	Información del número de referencia de la transacción
item_name	Retorna el nombre de la persona quien realizo la donación
cm	Retorna el espacio custom que contiene el comentario opcional
amt	Retorna el valor por el cual fue realizado la donación
st	Retorna el estado de la donación

Tabla 59 Respuesta del Webservice de PayPal
Fuente: Elaboración propia

5.6.5 Aplicación de las iteraciones

La metodología ágil de desarrollo implementada es Extreme Programming, la misma es detallada en la sección 5.1 de este capítulo. A continuación, se muestra cada una de las iteraciones, donde muestran principalmente la creación de los modelos, las vistas y controladores que son parte de la arquitectura MVC.

5.6.6 Iteración #1: Gestión de usuarios

Modelo de usuario

En la figura 29, se muestra el modelo creado para los usuarios.

```php
class User extends Authenticatable
{
    use HasFactory, Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name',
        'last_name',
        'email',
        'password',
        'status',
    ];

    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];

    /**
     * The attributes that should be cast to native types.
     *
     * @var array
     */
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];

    public function User_to_blog()
    {
        return $this->hasMany(Blog::class);
    }
}
```

Figura 29 Modelo de usuario
Fuente: Elaboración propia

Controlador de usuarios

En la figura 30, se muestra el controlador con los principales métodos para el funcionamiento de la gestión de usuarios que realiza el actor ACT-01.

```php
class UserController extends Controller
{
    public function index()
    {
        $data['new_users'] = User::query()
            ->select(DB::raw('count(*) as count'))
            ->where('created_at', '>=', Carbon::now()->firstOfMonth()->toDateTimeString())
            ->where('status', '=', '1')
            ->first();
        $data['active_users'] = User::query()
            ->select(DB::raw('count(*) as count'))
            ->where('status', '=', '1')
            ->first();
        $data['inactive_users'] = User::query()
            ->select(DB::raw('count(*) as count'))
            ->where('status', '=', '0')
            ->first();
        $data['all_users'] = User::query()
            ->select(DB::raw('count(*) as count'))
            ->first();
        return view('dashboard.user.user', $data);
    }

    public function store(Request $request)
    {
        $this->validate($request, [
            'name' => 'required',
            'last_name' => 'required',
            'email' => 'required',
            'password' => 'required',
        ]);

        return User::create([
            'name' => $request['name'],
            'last_name' => $request['last_name'],
            'email' => $request['email'],
            'password' => \Hash::make($request['password']),
            'status' => 1,
        ]);
    }

    public function update(Request $request, $id)
    {
        $this->validate($request, [
            'name' => 'required',
            'last_name' => 'required',
            'email' => 'required',
        ]);

        if ($request->has('password')) {
            $request['password'] = \Hash::make($request['password']);
        }

        $user = User::query()->findOrFail($id);
        $user->update($request->all());
    }

    public function destroy($id)
    {
        try {
            $user = User::findOrFail($id);
            if ($user->status == 1) {
                User::where('id', $id)->update(array('status' => '0'));
                return response()->json([
                    'message' => 'El usuario fue inactivado'
                ]);
            } else {
                User::where('id', $id)->update(array('status' => '1'));
                return response()->json([
                    'message' => 'El usuario fue activado nuevamente'
                ]);
            }

        } catch (Exception $exception) {
            Log::error($exception);
            return response()->make($exception->getMessage(), 500);
        }
    }

    public function getTable()
    {
        return User::query()->select('id', 'name', 'last_name', 'email', 'status')->get();
    }
```

Figura 30 Controlador de usuarios
Fuente: Elaboración propia

Inicio de sesión

En la figura 31 se muestra la vista para el ingreso de los usuarios para la aplicación de la ONG, la misma tiene los espacios para ingresar el correo junto con la contraseña, en caso de no existir o no coincidir la información ingresada, aparecerá un mensaje con el texto en rojo con la respuesta del servidor. Adicionalmente, hay una casilla para marcar si el usuario desea mantener la sesión abierta en el dispositivo para no volver a realizar ese paso de autenticación hasta que cierre la sesión.

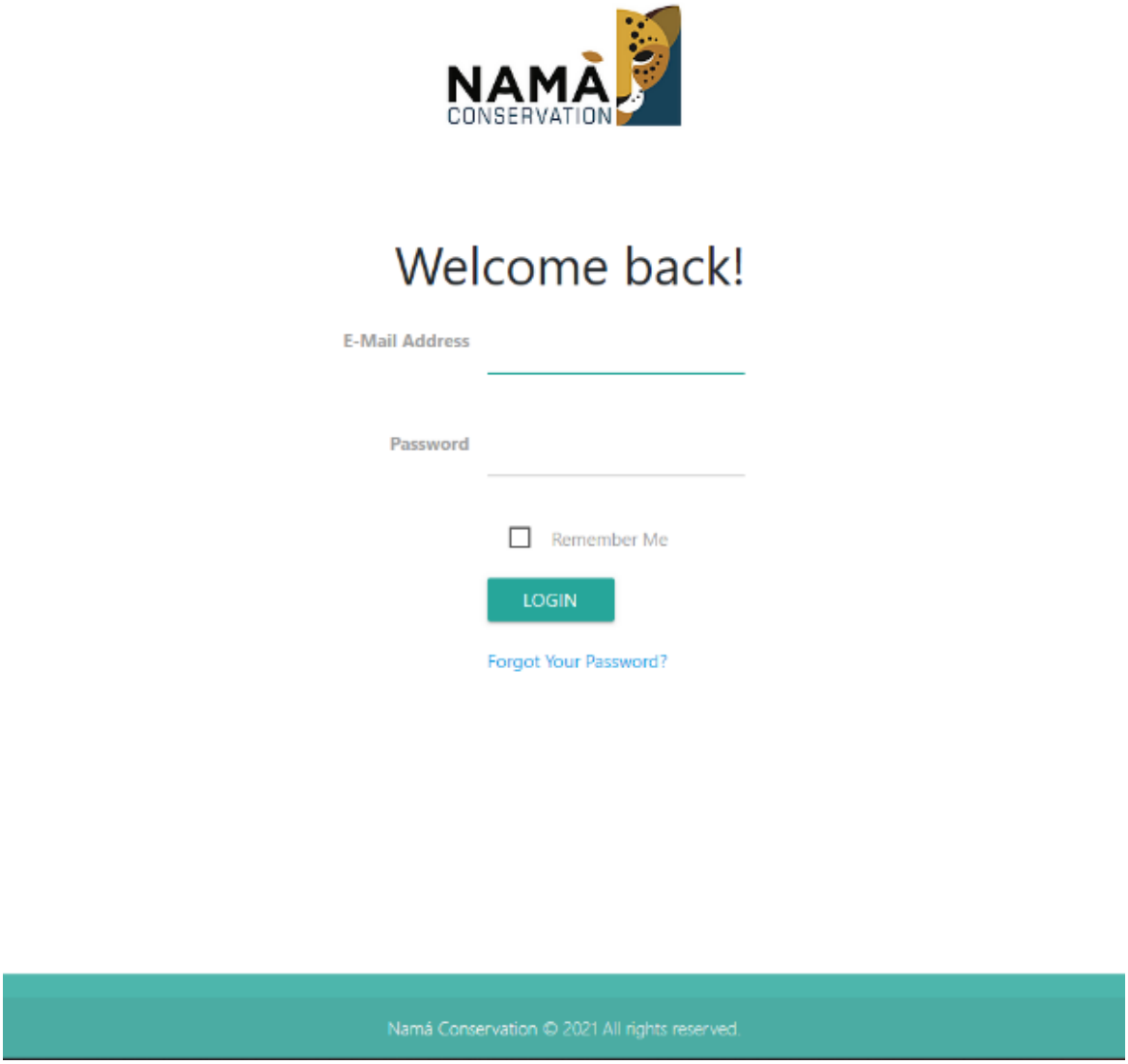


Figura 31 Pantalla de login
Fuente: Elaboración propia

Olvido de contraseña

La figura 32 muestra la vista para el olvido de la contraseña en el caso de que el usuario no recuerde la información. Existe una opción del login que redirige al usuario a esta vista, donde debe de ingresar el correo del usuario. En caso de que el correo exista dentro de la base de datos, la aplicación enviara un correo con un link para el cambio de la contraseña. En caso de no existir el correo ingresado, aparecerá un texto en color rojo con la respuesta del servidor.

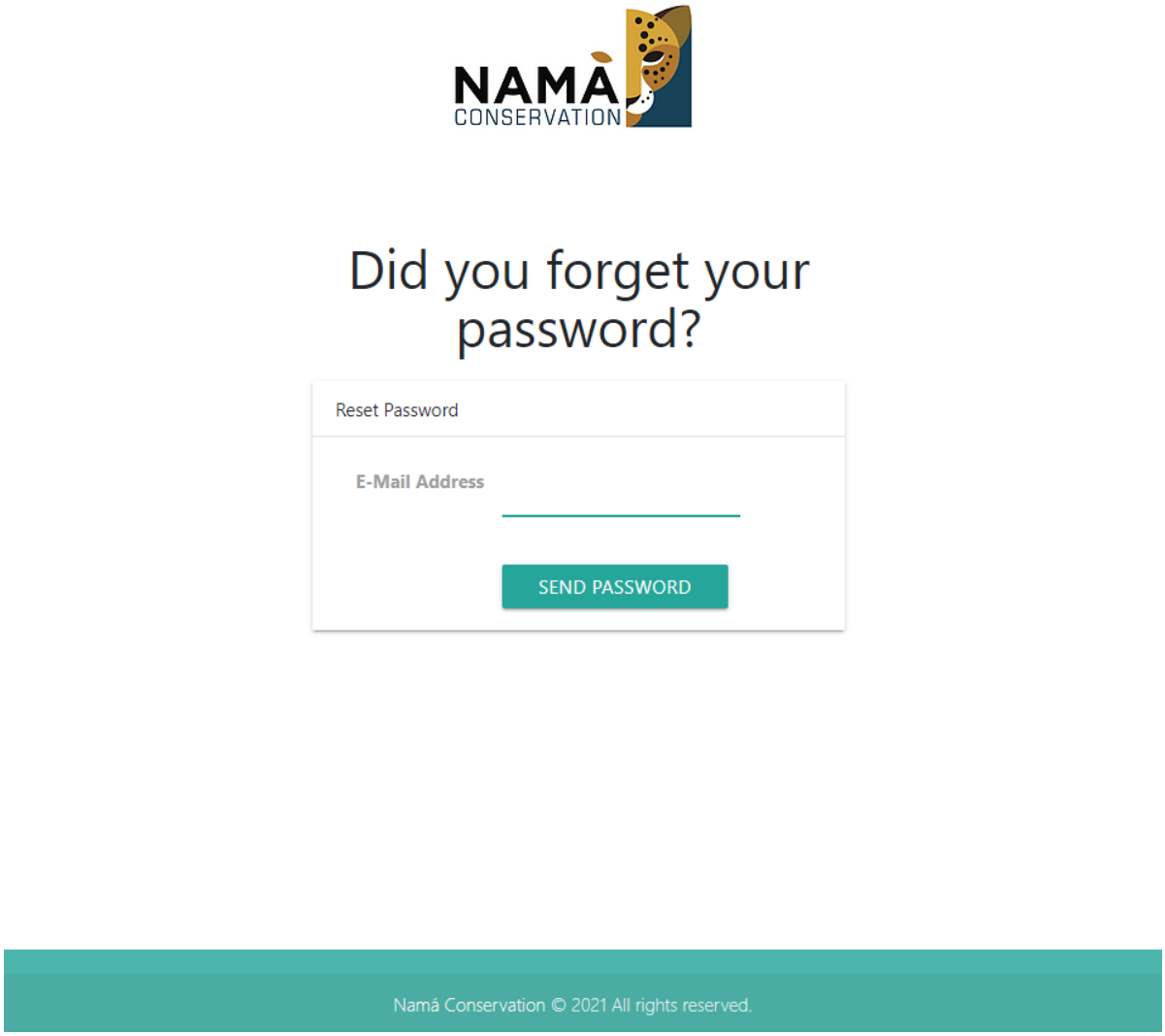


Figura 32 Pantalla de olvido de contraseña
Fuente: Elaboración propia

Reseteo de contraseña

En la figura 33 se muestra la vista para el reseteo de la contraseña, donde presenta 2 espacios para la contraseña, uno es para el cambio y el otro funciona para la confirmación de esta para asegurar de que el usuario realmente tenga el conocimiento de la nueva contraseña ingresada. El espacio del correo es llenado automáticamente desde la aplicación.

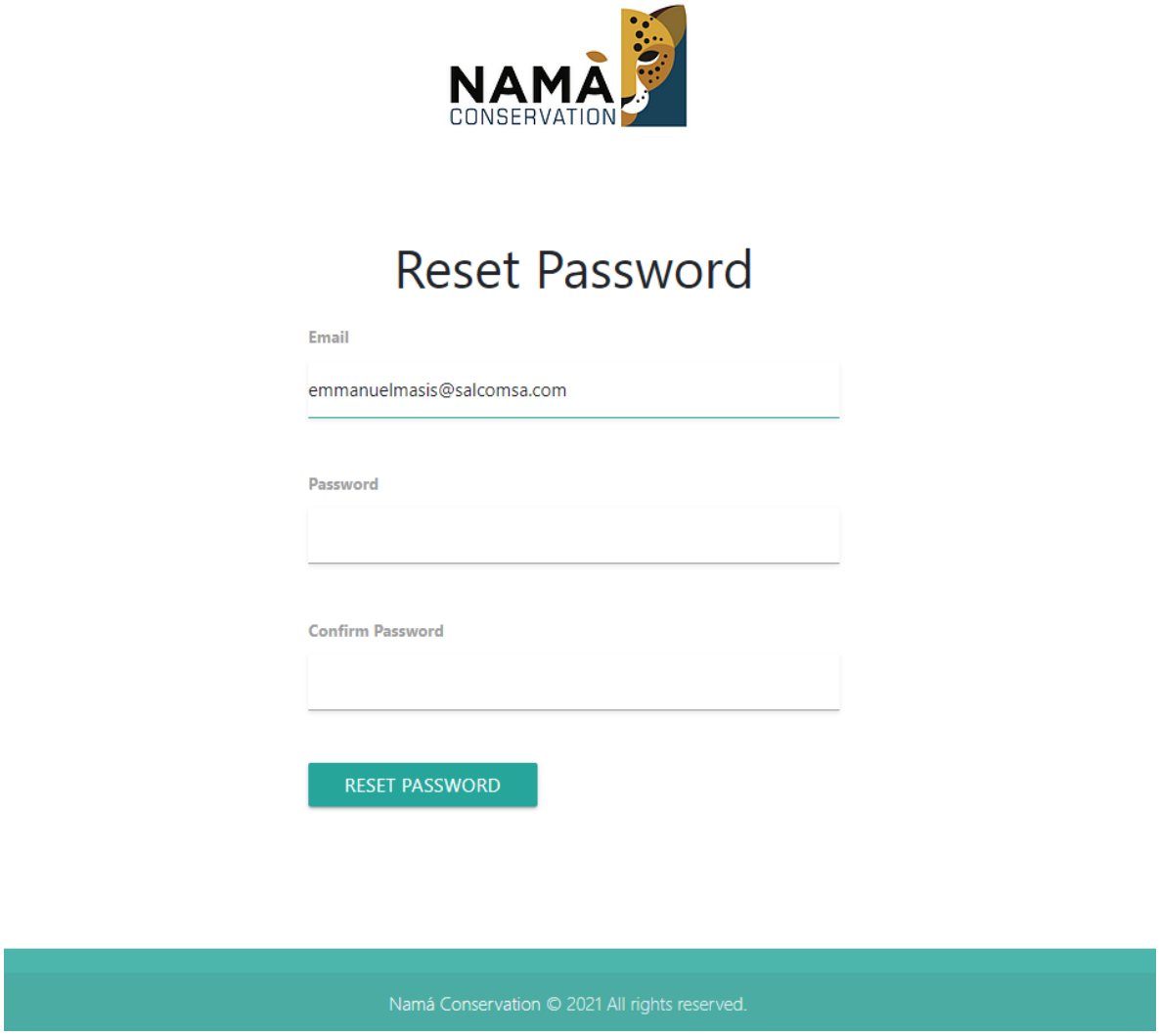


Figura 33 Pantalla de reseteo de contraseña
Fuente: Elaboración propia

Listado de usuarios

En la figura 34 se muestra la vista de los usuarios existentes del sistema, en la parte superior existe un filtro de busqueda para encontrar o filtrar el usuario deseado. Adicionalmente, en la parte superior derecha esta el boton para la creación de un usuario nuevo y dentro de la tabla en cada fila, esta la opción para editar o inactivar a algún usuario en especifico.

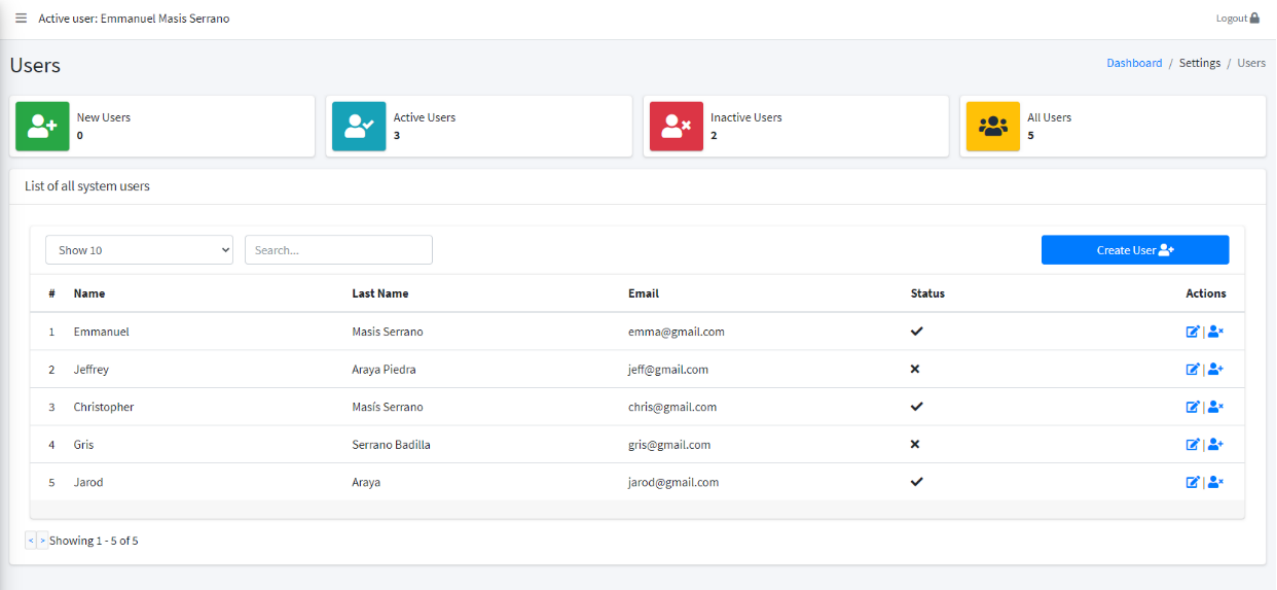


Figura 34 Pantalla de usuarios
Fuente: Elaboración propia

Formulario de creación de usuarios

En la figura 35 se muestra el formulario a llenar a la hora de crear un usuario nuevo. El mismo contiene los espacios solicitados en el requerimiento REQF-009

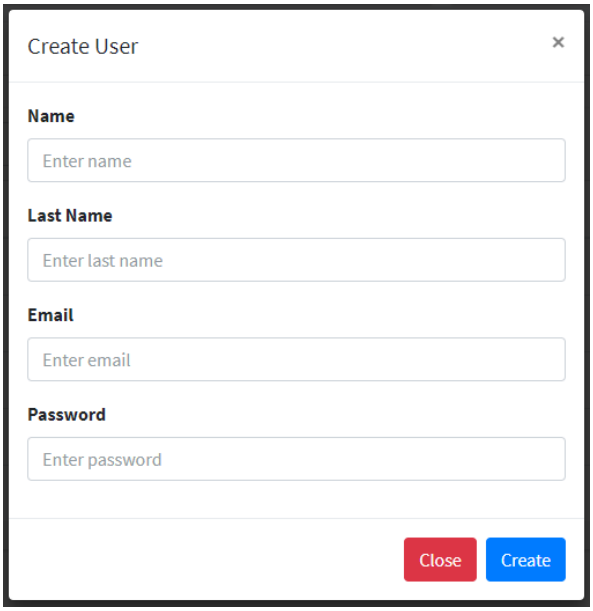


Figura 35 Ventana de creación de usuario
Fuente: Elaboración propia

Formulario de modificación de usuarios

En la figura 36 se muestra el formulario para la modificación de un usuario, el mismo a la hora seleccionar el usuario viene lleno para la actualización de los datos.

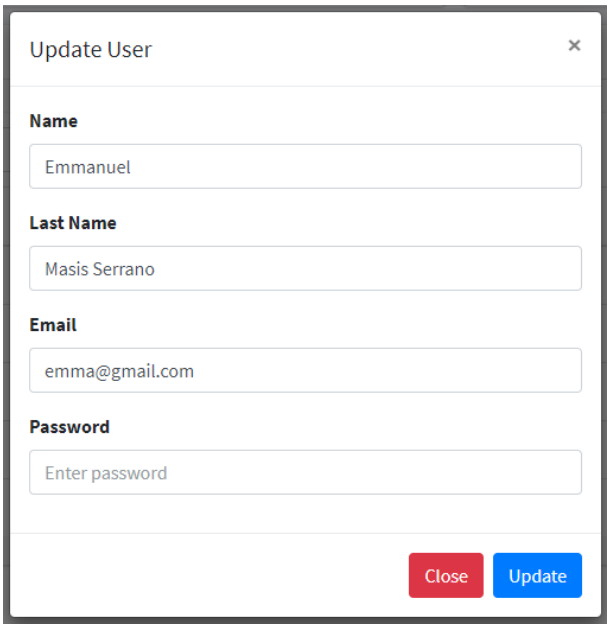


Figura 36 Ventana de modificación de usuario
Fuente: Elaboración propia

Ventanas de inactivación de usuarios

En la figura 34, se muestra una tabla donde en cada fila hay un icono de un usuario en la columna de Acciones, al seleccionar esa opción puede inactivar o activar al usuario según su estado actual. Para ello si el usuario esta activo y se desea pasar a inactivo, se muestra el mensaje de la figura 37. En caso de querer volver a activar a un usuario aparece la ventana de la figura 38.

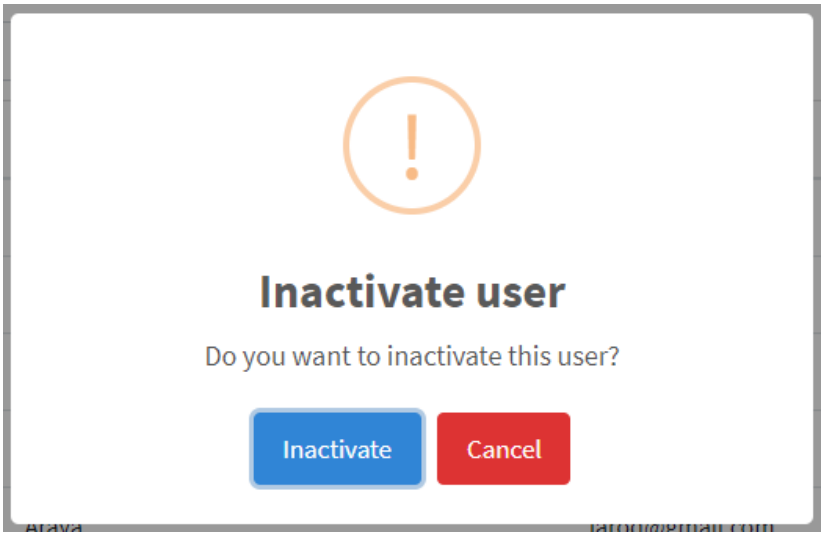


Figura 37 Ventana de inactivación de usuario
Fuente: Elaboración propia

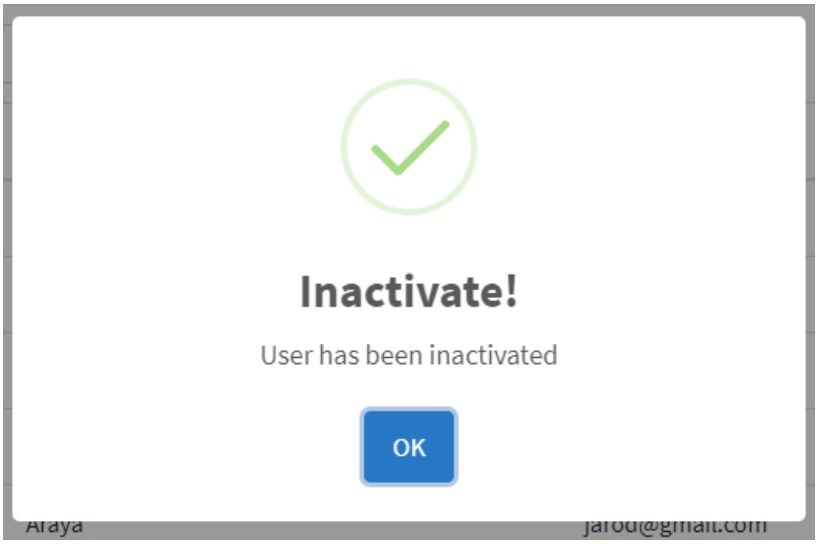


Figura 38 Ventana de confirmación de inactivación
Fuente: Elaboración propia

En caso de que el usuario inactivo tenga una sesión abierta, esta automáticamente se bloqueará y mostrará la figura 39.

Figura 39 Pantalla de acceso no autorizado
Fuente: Elaboración propia

5.6.7 Iteración #2: Gestión de intereses

Modelo de interés

En la figura 40 se muestra el modelo creado para los intereses.

```php
class Interest extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'interests';

    protected $fillable = ['description'];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];
}
```

Figura 40 Modelo de intereses
Fuente: Elaboración propia

Controlador de intereses

En la figura 41, se muestra el controlador con los principales métodos para el funcionamiento de la gestión de intereses.

```php
class InterestController extends Controller
{
    public function index()
    {
        return view('dashboard.interest.interest');
    }

    public function getTable()
    {
        return Interest::query()->get();
    }

    public function store(Request $request)
    {

        $this->validate($request, [
            'description' => 'required',
        ]);

        return Interest::create([
            'description' => $request['description'],
        ]);
    }

    public function update(Request $request, $id)
    {
        $this->validate($request, [
            'description' => 'required',
        ]);

        $data = Interest::query()->findOrFail($id);
        $data->update($request->all());
    }

    public function destroy($id)
    {
        try {
            $data = Interest::findOrFail($id);
            $data->delete();

        } catch (Exception $exception) {
            Log::error($exception);
            return response()->make($exception->getMessage(), 500);
        }
    }
}
```

Figura 41 Controlador de intereses
Fuente: Elaboración propia

Listado de intereses

En la figura 42 se muestra la vista para mostrar todos los intereses existentes del sistema, los mismos pueden ser filtrados por el buscador y pueden ser creados, modificados o eliminados.

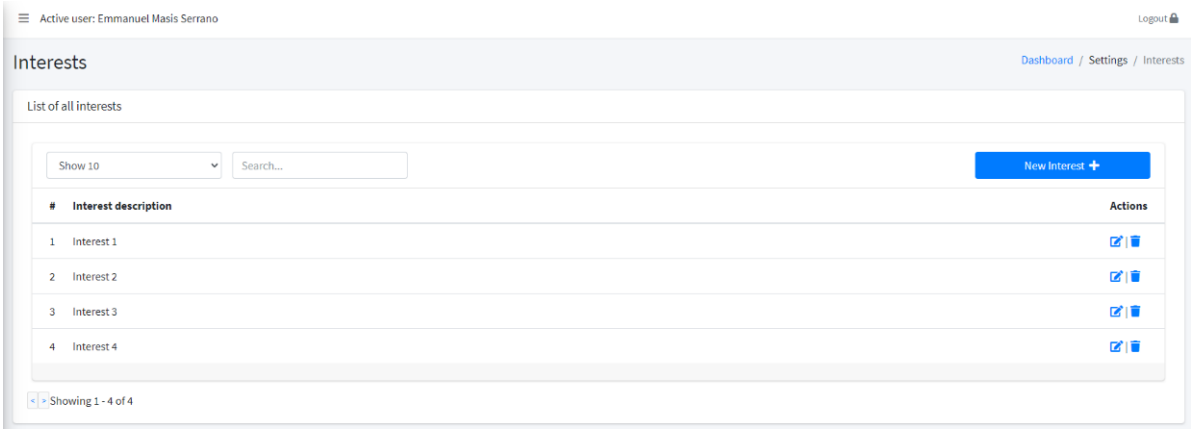


Figura 42 Pantalla de intereses
Fuente: Elaboración propia

Ventana de creación de interés

En la figura 43 se muestra la ventana que se despliega a la hora de crear un interés.

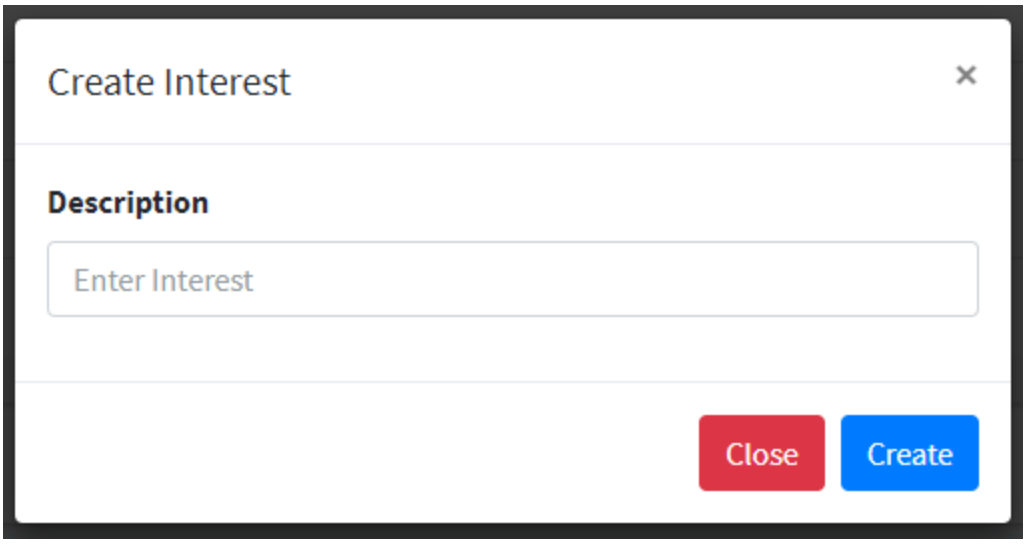


Figura 43 Ventana de creación de intereses
Fuente: Elaboración propia

Ventana de modificación de interés

En la figura 44 se muestra la ventana que se despliega a la hora de modificar un interés en específico.

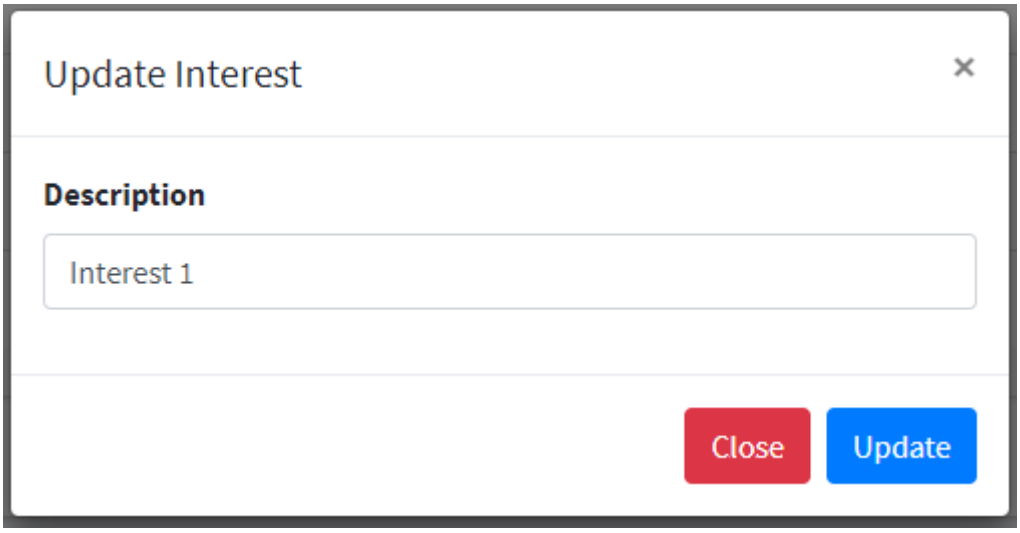


Figura 44 Ventana de modificación de intereses
Fuente: Elaboración propia

Ventanas de eliminación de interés

En la figura 45 se muestra la ventana de confirmación a la hora de eliminar un interés que fue seleccionado en la fila de la tabla de la figura 42.

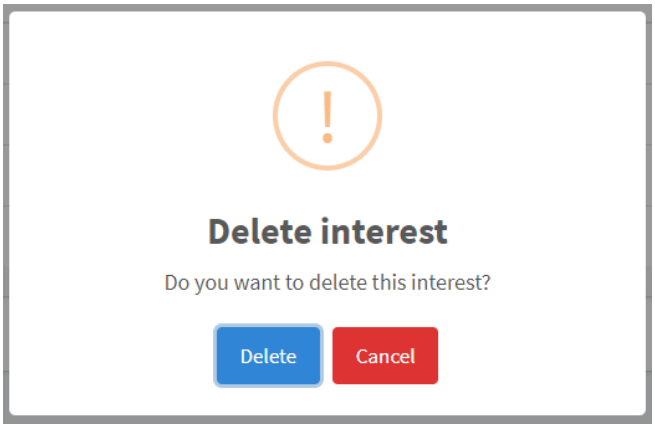


Figura 45 Ventana de eliminación de intereses
Fuente: Elaboración propia

Y en la figura 46 se muestra la ventana de confirmación a la hora de eliminar un interés.

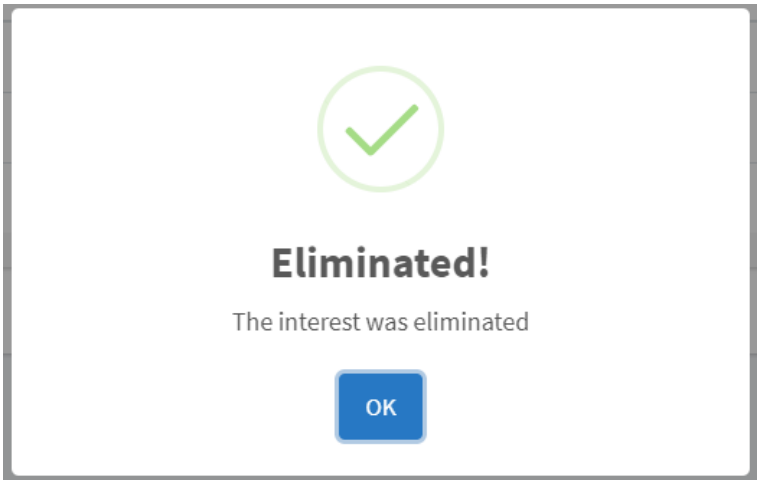


Figura 46 Ventana de confirmación de intereses eliminados
Fuente: Elaboración propia

5.6.8 Iteración #3: Gestión de programas

Modelo de programas

En la figura 47 se muestra el modelo creado para los programas.

```php
class Program extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'programs';

    protected $fillable = ['description'];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];
}
```

Figura 47 Modelo de programas
Fuente: Elaboración propia

Controlador de programas

En la figura 48, se muestra el controlador con los principales métodos para el funcionamiento de la gestión de programas.

```php
class ProgramController extends Controller
{
    public function index()
    {
        return view('dashboard.program.program');
    }

    public function getTable()
    {
        return Program::query()->get();
    }

    public function store(Request $request)
    {

        $this->validate($request, [
            'description' => 'required',
        ]);

        return Program::create([
            'description' => $request['description'],
        ]);
    }

    public function update(Request $request, $id)
    {
        $this->validate($request, [
            'description' => 'required',
        ]);

        $data = Program::query()->findOrFail($id);
        $data->update($request->all());
    }

    public function destroy($id)
    {
        try {
            $data = Program::findOrFail($id);
            $data->delete();

        } catch (Exception $exception) {
            Log::error($exception);
            return response()->make($exception->getMessage(), 500);
        }
    }
}
```

Figura 48 Controlador de programas
Fuente: Elaboración propia

Listado de programas

En la figura 49 se muestra la vista para mostrar todos los programas existentes del sistema, los mismos pueden ser filtrados por el buscador y pueden ser creados, modificados o eliminados.

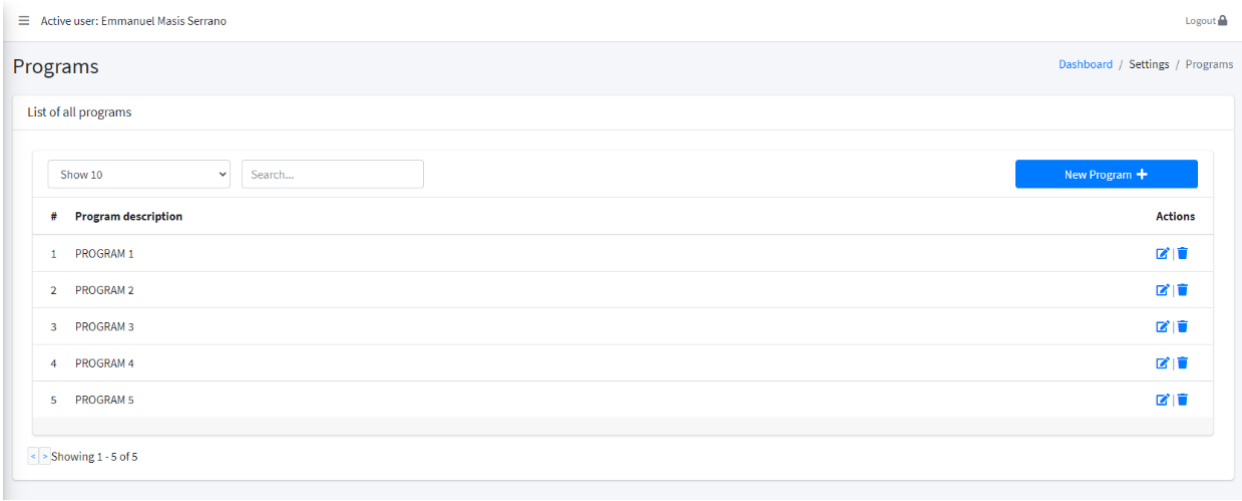


Figura 49 Pantalla de programas
Fuente: Elaboración propia

Ventana de creación de programa

En la figura 50 se muestra la ventana que se despliega a la hora de crear un programa.

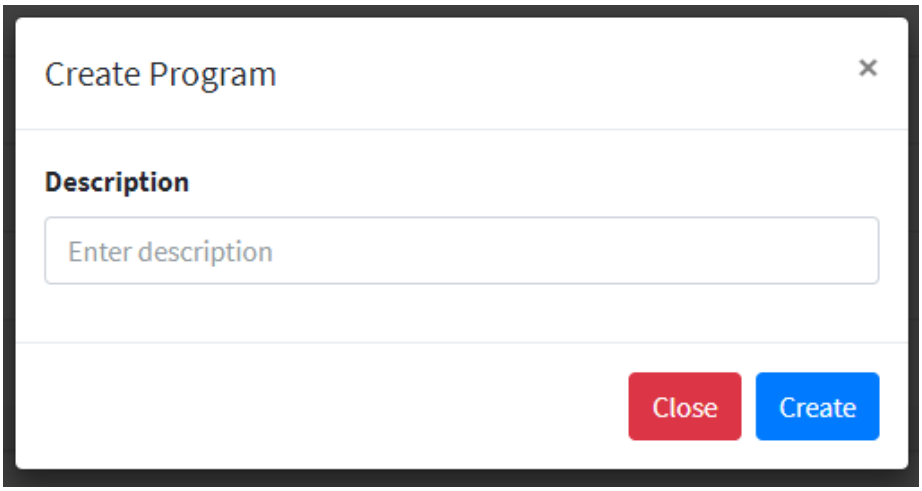


Figura 50 Ventana de creación de programas
Fuente: Elaboración propia

Ventana de modificación de programa

En la figura 51 se muestra la ventana que se despliega a la hora de modificar un programa en específico.

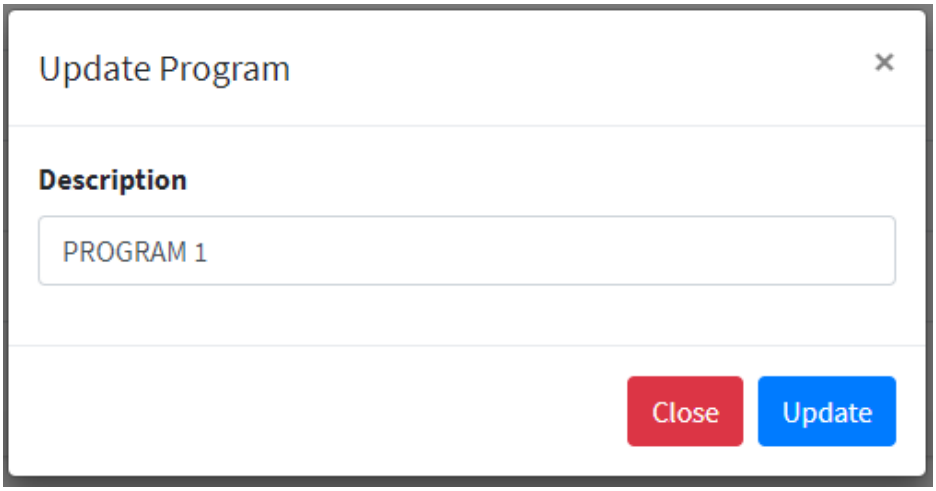


Figura 51 Ventana de modificación de programas
Fuente: Elaboración propia

Ventanas de eliminación de programas

En la figura 52 se muestra la ventana de confirmación a la hora de eliminar un programa que fue seleccionado en la fila de la tabla de la figura 49.

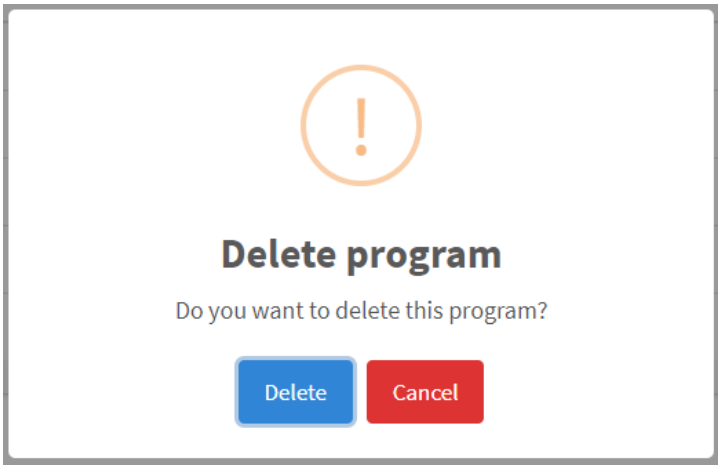


Figura 52 Ventana de eliminación de programas
Fuente: Elaboración propia

Y en la figura 53 se muestra la ventana de confirmación a la hora de eliminar un programa.

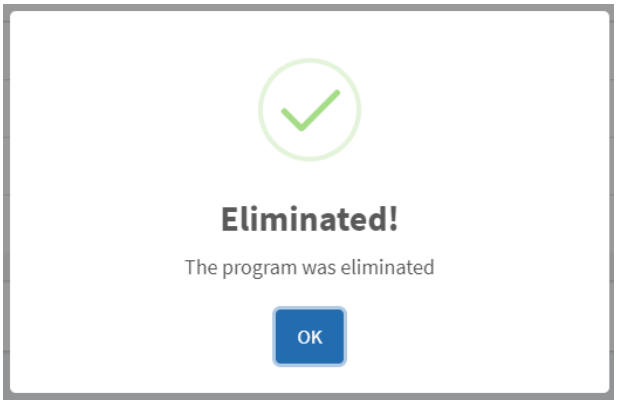


Figura 53 Ventana de confirmación de programas eliminados
Fuente: Elaboración propia

5.6.9 Iteración #4: Módulo de voluntariado

Modelos de voluntariados

En la figura 54 se muestra el modelo creado para los voluntarios.

```php
class Volunteer extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'volunteers';

    protected $fillable = [
        'name',
        'last_name',
        'email',
        'birthday',
        'nationality_id',
        'expectations',
        'skills',
        'created_at'
    ];

    protected $hidden = ["updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];

    public function nationality()
    {
        return $this->belongsTo(Nationality::class, 'nationality_id');
    }

    protected function serializeDate(DateTimeInterface $date)
    {
        return $date->format('Y-m-d H:i:s');
    }

}
```

Figura 54 Modelo de voluntariado
Fuente: Elaboración propia

En la figura 55 se muestra el modelo VolunteerAddress, este modelo es para el almacenamiento de las direcciones de los voluntarios.

```php
class VolunteerAddress extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'volunteer_addresses';

    protected $fillable = [
        'volunteer_id',
        'country_id',
        'city',
        'address',
    ];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];

}
```

Figura 55 Modelo de dirección de voluntarios
Fuente: Elaboración propia

La figura 56 muestra el modelo de país, este modelo funciona para optimizar la base de datos para que no exista información duplicada.

```php
class Country extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'countries';

    protected $fillable = ['name'];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];
}
```

Figura 56 Modelo de países
Fuente: Elaboración propia

En la figura 57 muestra el modelo VolunteerInterest, este modelo funciona para almacenar los diferentes intereses que el voluntario puede tener a la hora de realizar un voluntariado.

```php
class VolunteerInterest extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'interest_by_volunteer';

    protected $fillable = [
        'interest_id',
        'volunteer_id',
    ];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];
}
```

Figura 57 Modelo de intereses de voluntarios
Fuente: Elaboración propia

La figura 58 muestra el modelo VolunteerProgram, este modelo funciona para almacenar los diferentes programas en que el voluntario puede participar.

```php
class VolunteerProgram extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'program_by_volunteer';

    protected $fillable = [
        'program_id',
        'volunteer_id',
    ];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];
}
```

Figura 58 Modelo de programas de voluntarios
Fuente: Elaboración propia

La figura 59 muestra el modelo de las nacionalidades, este modelo funciona para optimizar la base de datos para que no existan nacionalidades repetidas.

```php
class Nationality extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'nationalities';

    protected $fillable = ['description'];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];
}
```

Figura 59 Modelo de nacionalidades
Fuente: Elaboración propia

Controlador de voluntariado

En la figura 60, se muestra el controlador con los principales métodos para el funcionamiento de la gestión de voluntariado para el actor ACT-01.

```php
class VolunteerReportController extends Controller
{
    public function index()
    {
        return view('dashboard.volunteer.volunteer');
    }

    public function getTable()
    {
        return Volunteer::with('nationality')->get();
    }

    public function destroy($id)
    {
        try {

            VolunteerAddress::query()->where('volunteer_id', '=', $id)->delete();
            VolunteerInterest::query()->where('volunteer_id', '=', $id)->delete();
            VolunteerProgram::query()->where('volunteer_id', '=', $id)->delete();
            $volunteer = Volunteer::findOrFail($id);
            $volunteer->delete();

        } catch (Exception $exception) {
            Log::error($exception);
            return response()->make($exception->getMessage(), 500);
        }
    }

    public function exportExcel($init_date, $end_date)
    {
        $init = $init_date.' 00:00:00';
        $end = $end_date.' 23:59:00';
        return \Excel::download(new VolunteersExport($init, $end), 'volunteers.xlsx');
    }
}
```

Figura 60 Controlador de voluntariado
Fuente: Elaboración propia

Formulario de creación de voluntarios

La figura 61 muestra el formulario de ingreso de información para realizar el registro de voluntarios, este formulario es parte de la página web de Namá Conservation donde cualquier actor ACT-02 puede registrarse sin necesidad de autenticación para ayudar al conservatorio en algún programa en específico. Todo el formulario tiene validaciones en JavaScript para determinar que el usuario ingrese la información correctamente. Además, el formulario fue creado con los campos solicitados en el requerimiento REQF-005 y es en esta sección donde se implementa la selección de los programas e intereses para que el personal de Namá pueda realizar la selección de los voluntarios según las necesidades.

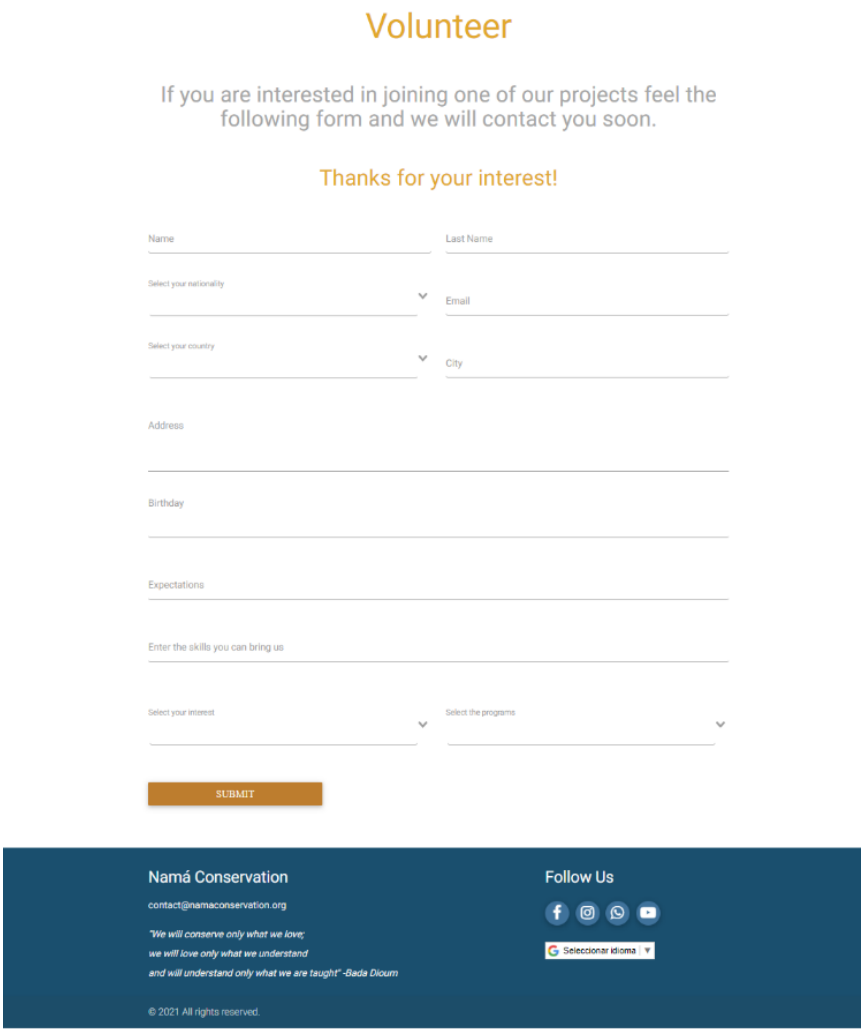


Figura 61 Pantalla de ingreso de voluntarios
Fuente: Elaboración propia

Listado de voluntariado

En la figura 62 se muestra la pantalla de listado de voluntariado, en esta pantalla se puede buscar, mostrar o eliminar un voluntario en específico. Además, puede generar el reporte de voluntariado según las fechas que el usuario ingrese en los filtros.

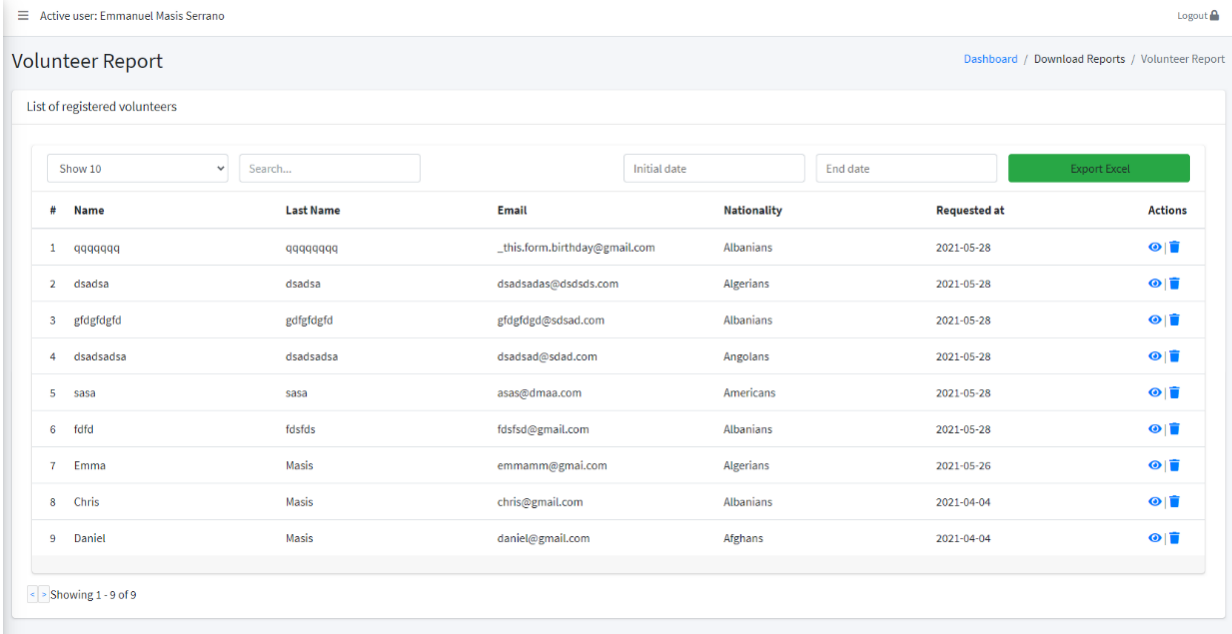


Figura 62 Pantalla de voluntariado
Fuente: Elaboración propia

Visualización de voluntario

En la figura 63 se muestra la ventana de visualización a la hora que se selecciona un registro de la tabla de la figura 62.

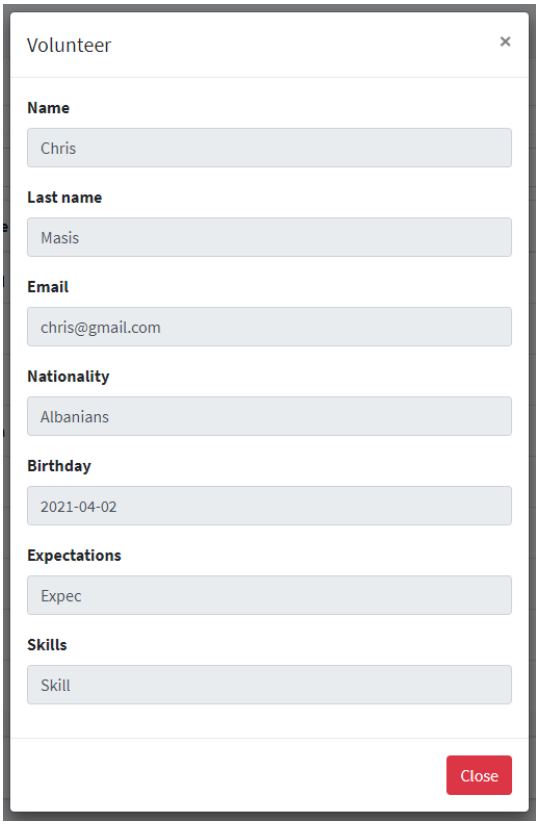


Figura 63 Ventana de visualización de voluntario
Fuente: Elaboración propia

Ventanas de eliminación de voluntarios

En la figura 64 se muestra la ventana de confirmación a la hora de eliminar un voluntario que fue seleccionado en la fila de la tabla de la figura 62.

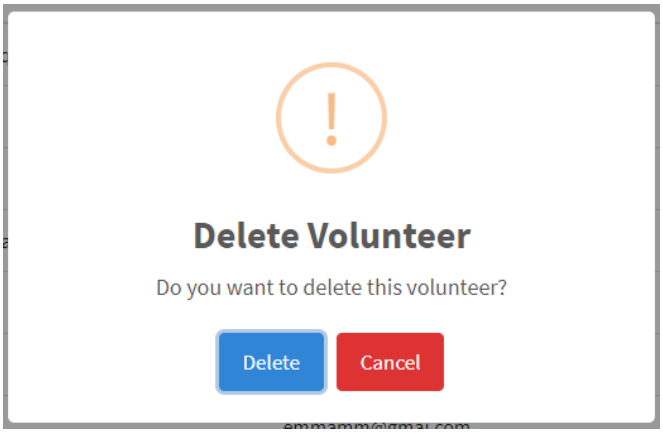


Figura 64 Ventana de eliminación de voluntario
Fuente: Elaboración propia

Reporte descargable de voluntariado

En la figura 65 se muestra la ventana de confirmación a la hora de realizar la exportación del reporte descargable en Excel de voluntariado. Para poder realizarlo se debe ingresar el rango de fechas a obtener la información y oprimir el botón de “Exportar Excel”.

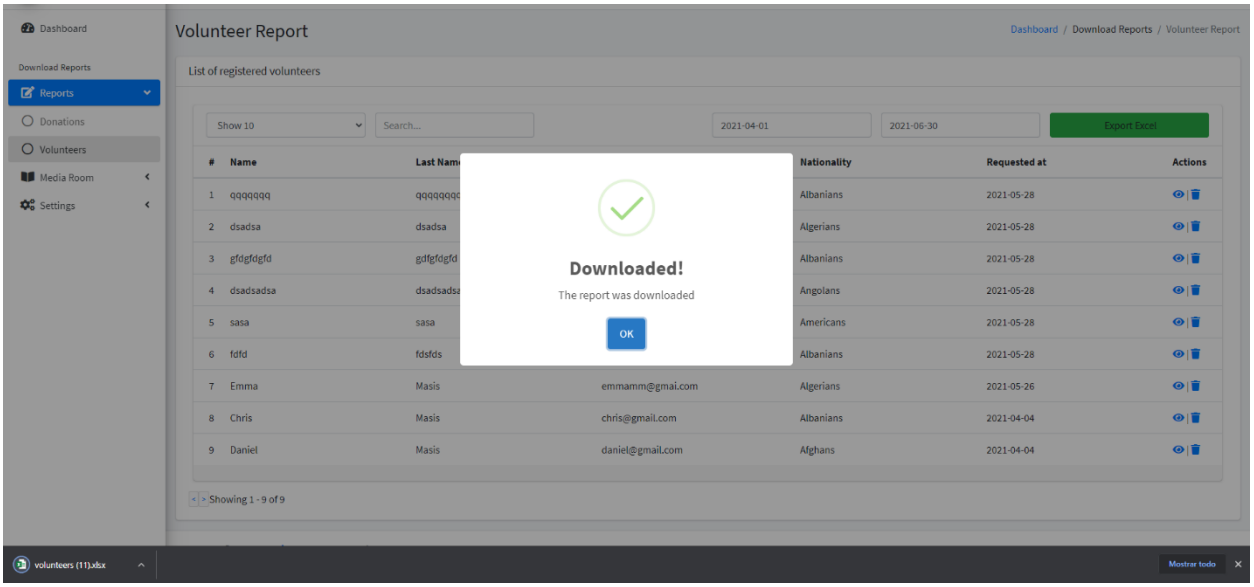


Figura 65 Pantalla de descarga de Excel de voluntariado
Fuente: Elaboración propia

Formato del reporte descargable de voluntariado

En la figura 66 se muestra el formato del reporte descargado tal y como se solicita en el requerimiento REQF-007.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Name	Last Name	Email	Nationality	Country	City	Address	Birthday	Expectations	Skills	Interests	Programs	Requested At
2	Chris	Masis	chris@gmail.com	Albanians	Afghanistan	Cartago	Prueba	2021-04-02	Expec	Skill	Interest 1	PROGRAM 4	2021-04-04 00:43:11
3	Daniel	Masis	daniel@gmail.com	Afghans	Algeria	Hola	Direccion	2021-04-09	Expect 2	Hola	Interest 2,Interest 2,Interest 2	PROGRAM 4,PROGRAM 1,PROGRAM 2	2021-04-04 00:43:11
4	Emma	Masis	emmamm@gmai.com	Algerians	Algeria	Coro	Coro	2021-05-12	Exp	Skill	Interest 2	PROGRAM 2	2021-05-26 03:29:51
5													

Figura 66 Ejemplo de reporte exportable de voluntariado
Fuente: Elaboración propia

5.6.10 Iteración #5: Módulo de publicaciones

Modelo de publicaciones

La figura 67 muestra el modelo de las publicaciones.

```php
class Blog extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'blog';

    protected $fillable = ['title', 'body', 'url_video', 'user_id', "created_at", "updated_at"];

    protected $hidden = ["deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];

    public function Blog_by_user()
    {
        return $this->belongsTo(User::class, 'id');
    }

    public function Blog_to_picture()
    {
        return $this->hasMany(BlogPicture::class, 'id');
    }

    protected function serializeDate(DateTimeInterface $date)
    {
        return $date->format('Y-m-d');
    }

}
```

Figura 67 Modelo de publicaciones.
Fuente: Elaboración propia

La figura 68 muestra el modelo para que pueda cargar las diferentes fotografías para cada publicación

```php
class BlogPicture extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'blog_pictures';

    protected $fillable = ['photo', 'blog_id'];

    protected $hidden = ["created_at", "updated_at", "deleted_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];

    public function Picture_by_blog()
    {
        return $this->belongsTo(Blog::class);
    }
}
```

Figura 68 Modelo de fotografías de publicaciones
Fuente: Elaboración propia

Controlador para la carga de las fotografías

La figura 69 muestra el controlador para cargar las fotografías de cada publicación dentro del servidor.

```php
class UploadImagesController extends Controller
{
    public function show($id)
    {
        $blog = Blog::find($id);
        $blog['pictures'] = BlogPicture::query()
            ->select()
            ->where('blog_pictures.blog_id', '=', $id)
            ->get();

        return view('dashboard.media.upload_images', compact('blog'));

    }

    public function store(Request $request)
    {
        $uuid = (string)Str::uuid();
        $imageName = $uuid . time() . '.' . $request->file->getClientOriginalExtension();
        $request->file->move(public_path('blog'), $imageName);

        $imageUpload = new BlogPicture();
        $imageUpload->photo = $imageName;
        $imageUpload->blog_id = $request->blog_id;
        $imageUpload->save();

        return response()->json(['success' => 'The image was uploaded successfully']);
    }

    public function destroy($name)
    {
        try {
            $data = BlogPicture::query()
                ->select()
                ->where('blog_pictures.photo', '=', $name)
                ->first();

            $data->delete();

            return response()->json([
                'message' => 'It was successfully eliminated'
            ]);
        } catch (Exception $e) {
            Log::error($e);
            return response()->make($e->getMessage(), 500);
        }
    }

}
```

Figura 69 Controlador de carga de imágenes
Fuente: Elaboración propia

Listado de publicaciones

En la figura 70 se muestra el listado de todas las publicaciones realizadas por los usuarios. Para poder crear una nueva publicación se debe oprimir en la opción de crear publicación. Para poder modificar, eliminar o cargar fotografías para la publicación se debe de posicionar en la fila de cada registro y seleccionar la opción a realizar en la columna de acción.

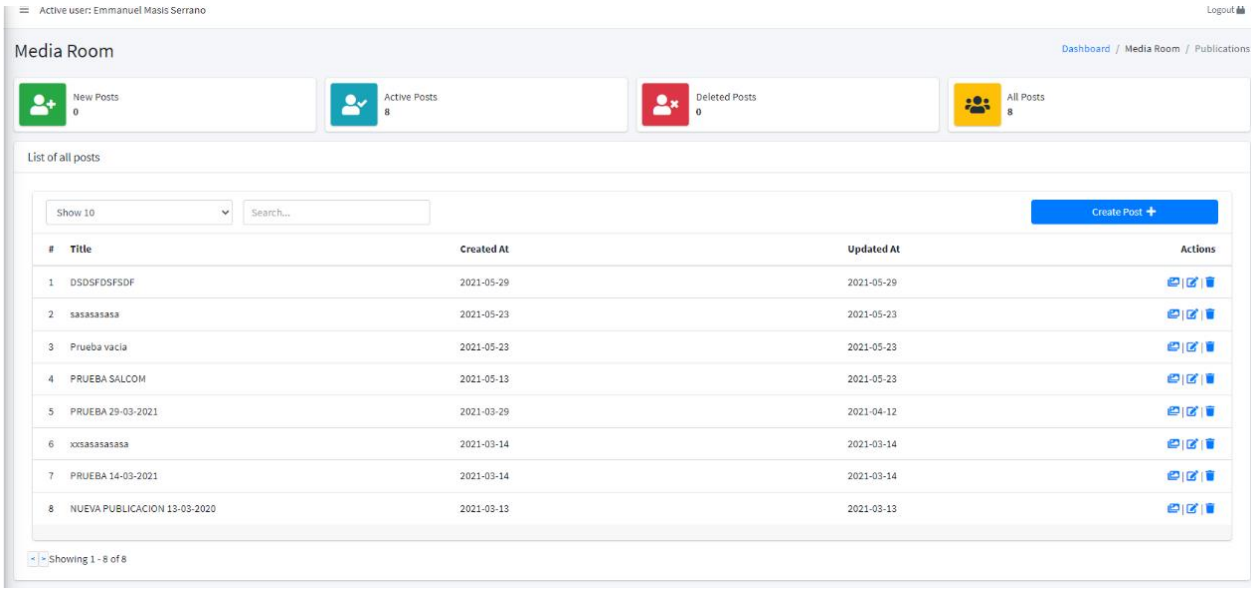


Figura 70 Pantalla de publicaciones
Fuente: Elaboración propia

Pantalla de creación de publicación

En la figura 71 se muestra la pantalla para la creación de la publicación. En esta vista se crea el título de la publicación, sucesivamente del texto o información a mostrar al actor ACT-02. Para el caso de las modificaciones se utiliza la misma pantalla solo que con la información con la que fue creada.

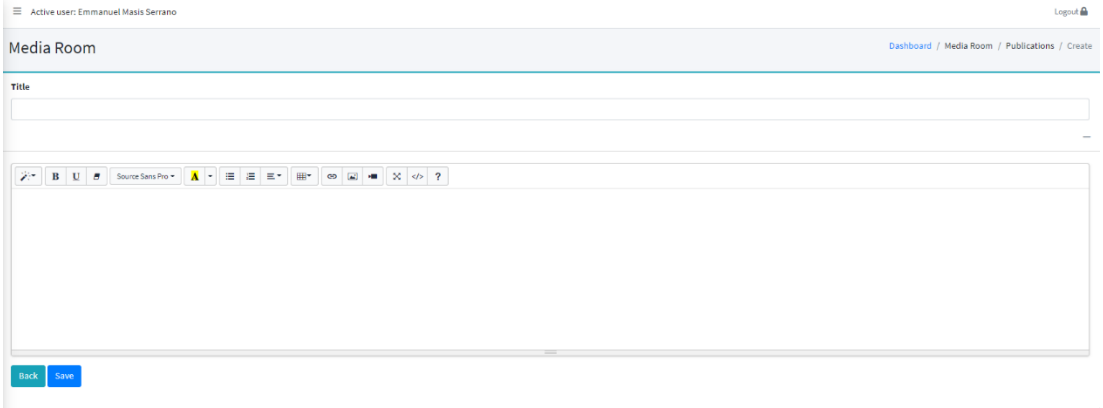


Figura 71 Pantalla de creación de publicaciones
Fuente: Elaboración propia

Pantalla de carga de imágenes para la publicación

En la figura 72 se muestra la pantalla para la carga de fotografías. Esta pantalla se muestra una vez creada la publicación, en ella solo se arrastran las imágenes dentro del cuadro principal para que estas sean almacenadas en el servidor. Las mismas pueden ser eliminadas si se posiciona el cursor encima de la foto y oprimiendo la opción de eliminar.

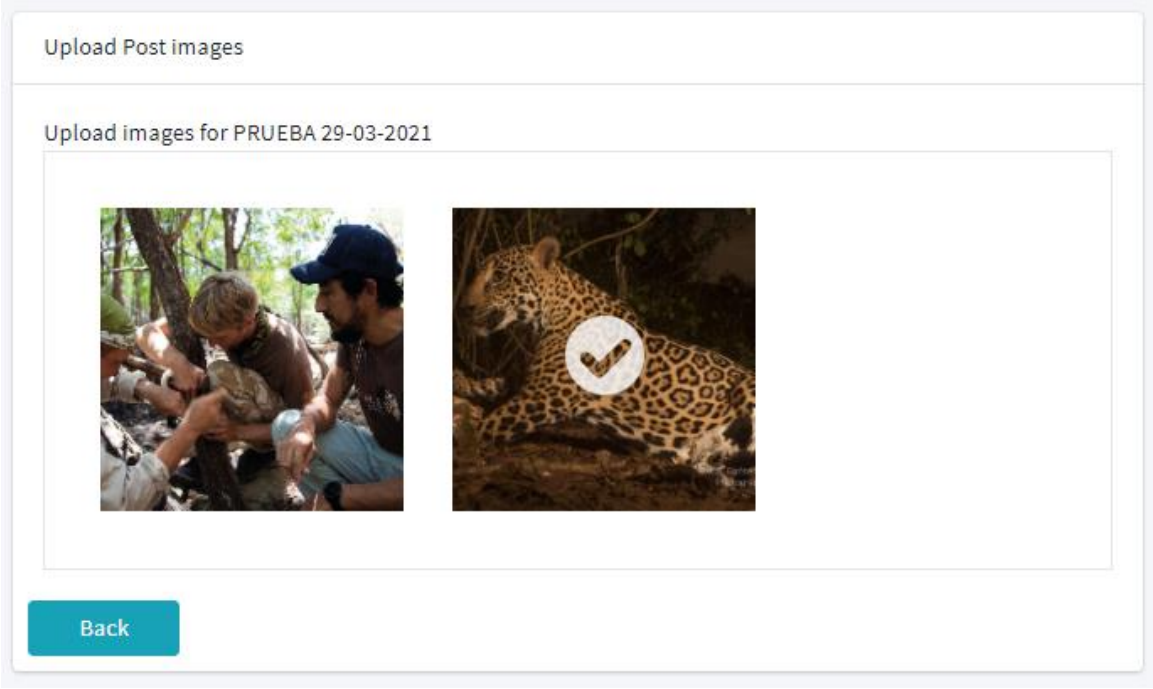


Figura 72 Pantalla de carga de fotografías
Fuente: Elaboración propia

Ventanas de eliminación de publicación

En la figura 73 se muestra el mensaje de confirmación para la eliminación de la publicación. La mismas son eliminadas desde la tabla que se muestran en la figura 70, para ello solo debe seleccionar la fila donde se encuentra el registro, seleccionar el icono con basurero que aparece en la columna de acciones.

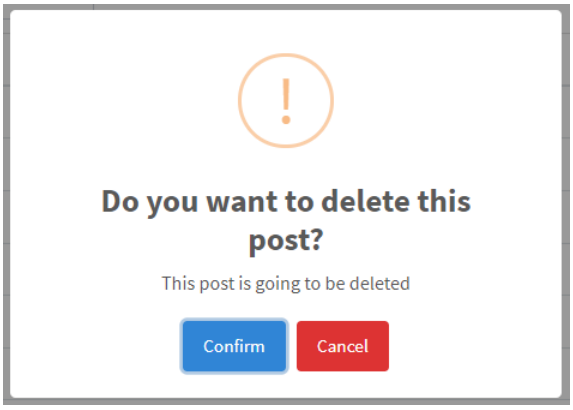


Figura 73 Ventana de eliminación de publicaciones
Fuente: Elaboración propia

Pantalla de visualización de publicación

En la figura 74 se muestra un ejemplo de la pantalla de la visualización que tendría el actor ACT-02 a la hora de ingresar a la página web de Nama Conservation y ver una de las publicaciones.

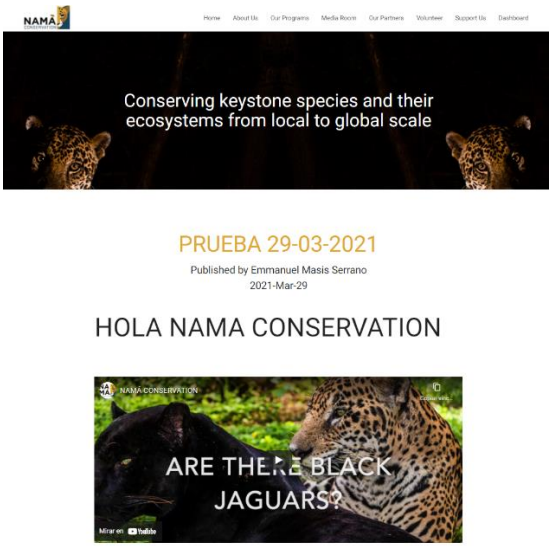


Figura 74 Pantalla de visualización de publicación
Fuente: Elaboración propia

5.6.11 Iteración #6: Módulo de donaciones

Modelo de donaciones

La figura 75 muestra el modelo de las donaciones.

```php
class Donation extends Model
{
    use HasFactory;
    use SoftDeletes;

    protected $table = 'donations';

    protected $fillable = ['identification', 'description', 'comment', 'amount', 'status', "created_at"];

    protected $dates = ["created_at", "updated_at", "deleted_at"];

    protected function serializeDate(DateTimeInterface $date)
    {
        return $date->format('Y-m-d H:i:s');
    }

}
```

Figura 75 Modelo de donaciones
Fuente: Elaboración propia

Controlador de donaciones

La figura 76 muestra el controlador de las donaciones con los principales métodos para el ACT-01.

```php
class DonationsReportController extends Controller
{
    public function index()
    {
        return view('dashboard.donations.donations');
    }

    public function getTable()
    {
        return Donation::query()->get();
    }

    public function exportExcel($init_date, $end_date)
    {
        $init = $init_date.' 00:00:00';
        $end = $end_date.' 23:59:00';
        return \Excel::download(new DonationsExport($init, $end), 'donations.xlsx');
    }

    public function destroy($id)
    {
        try {
            $data = Donation::findOrFail($id);
            $data->delete();

        } catch (Exception $exception) {
            Log::error($exception);
            return response()->make($exception->getMessage(), 500);
        }
    }

}
```

Figura 76 Controlador de donaciones
Fuente: Elaboración propia

Pantalla de donaciones

En la figura 77 se muestra la pantalla del formulario para cuando el actor ACT-02 desea realizar una donación a Namá Conservation. Las misma se encuentra dentro de la página web. En esta se solicita el nombre de la persona, organización o un alias en caso de realizar una donación anónima, un espacio opcional por si el ACT-02 desea realizar un comentario y el monto a donar.

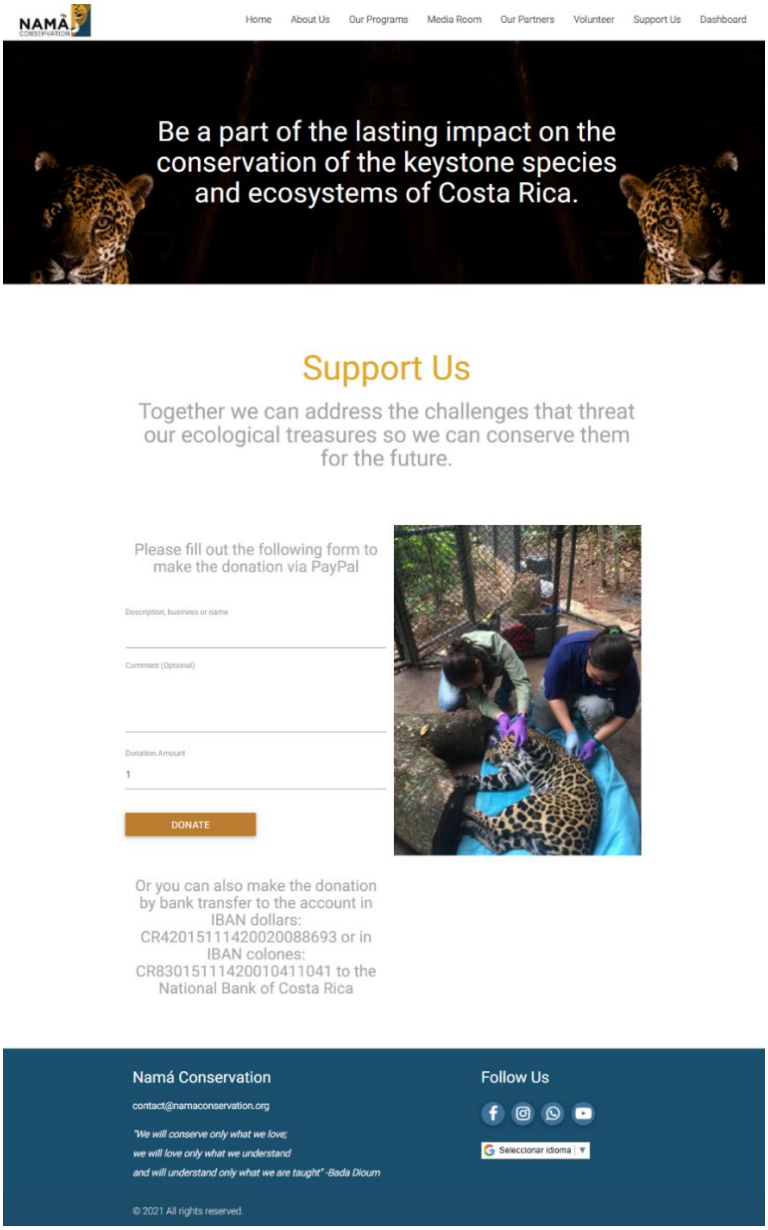


Figura 77 Pantalla de creación de donación
Fuente: Elaboración propia

Pantalla de selección de pago

En la figura 78 se muestra la pantalla que brinda PayPal para la selección del pago de la donación. Si la opción seleccionada es “Donar con PayPal” el ACT-02 deberá realizar el login de la cuenta de PayPal y seleccionar el pago según los métodos que tenga registrado. Caso contrario sino cuenta con cuenta de PayPal puede seleccionar la opción de “pago con tarjeta de débito o crédito” donde deberá llenar los datos de la figura 79 para poder realizar la donación.

Figura 78 Pantalla de selección de pago
Fuente: PayPal

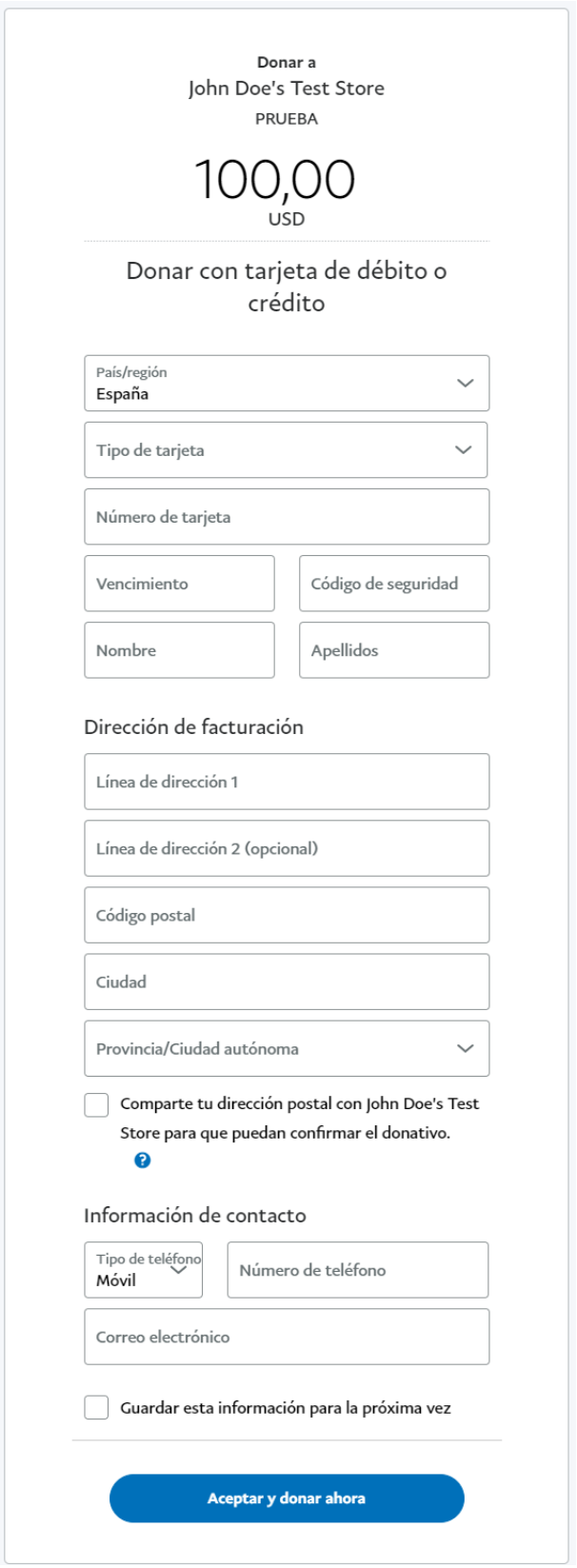


Figura 79 Pantalla de formulario para donación con tarjeta bancaria
Fuente: PayPal

Pantalla de confirmación de donación de PayPal

En la figura 80 se muestra la confirmación de PayPal una vez que la donación fue realizada satisfactoriamente. A los segundos esta vista se redirige a la pantalla de agradecimiento de Namá Conservation como se muestra en la figura 81.

Figura 80 Pantalla de confirmación de PayPal
Fuente: PayPal

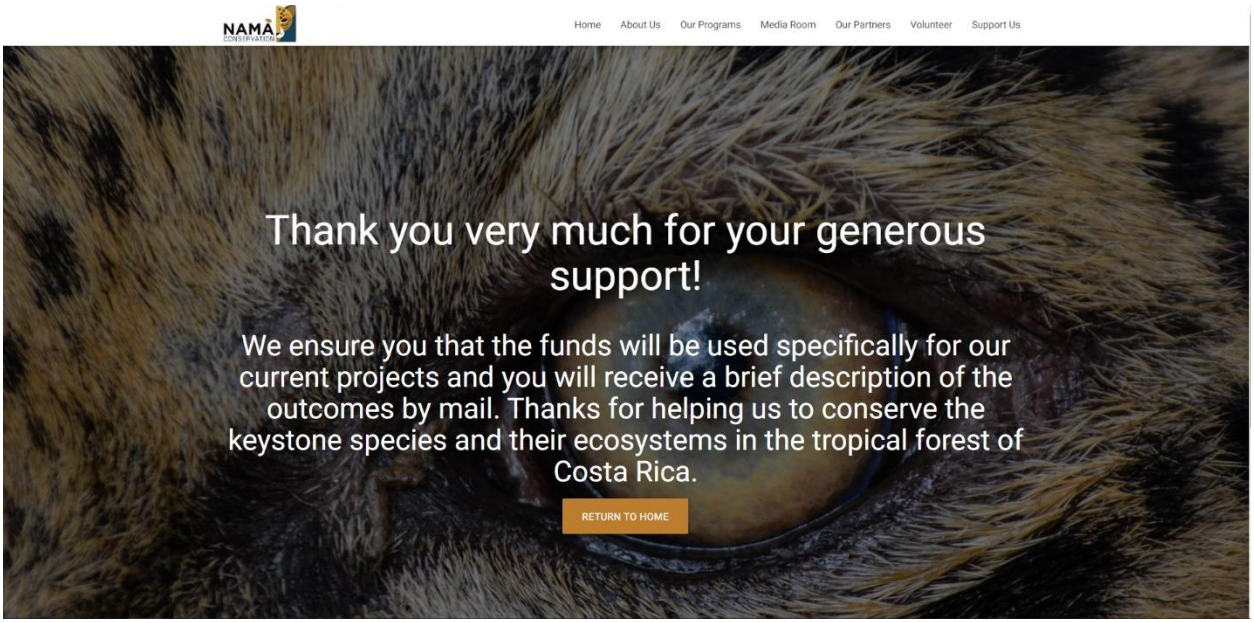


Figura 81 Pantalla de agradecimiento de donación
Fuente: Elaboración propia

Listado de donaciones

Una vez que se realiza la donación, esta es registrada en la aplicación web. En la figura 82 se muestra el listado de las donaciones. Donde el actor ACT-01 puede filtrar, buscar, visualizar, eliminar y descargar la información de la donación realizada.

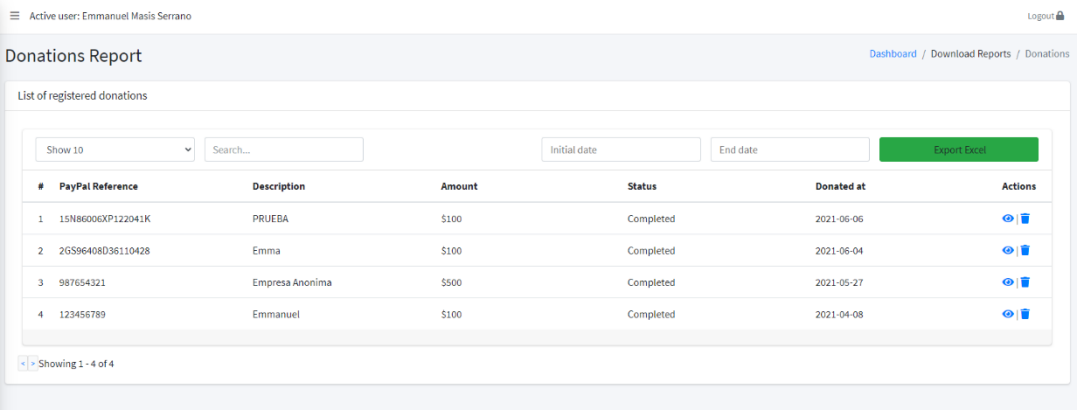


Figura 82 Pantalla de donaciones
Fuente: Elaboración propia

Ventana de visualización de donación

En la figura 83 se muestra la ventana que muestra el desglose de la donación una vez que es seleccionada de la tabla de la figura 80.

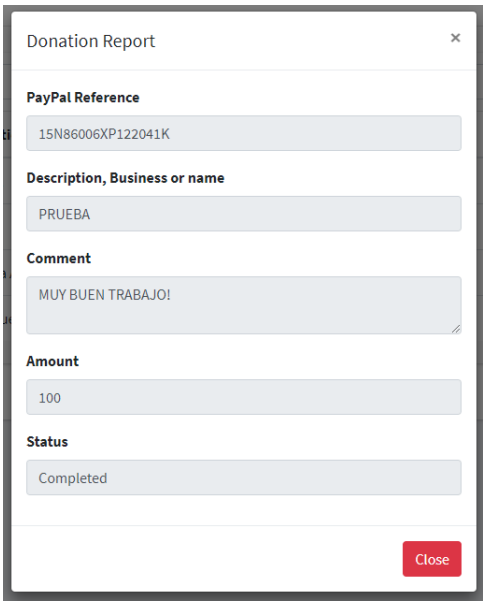


Figura 83 Ventana de visualización de donación
Fuente: Elaboración propia

Ventana de eliminación de donación

En la figura 84 se muestra el mensaje de confirmación para la eliminación de la donación. La mismas son eliminadas desde la tabla que se muestran en la figura 80, para ello solo debe seleccionar la fila donde se encuentra el registro, seleccionar el icono con basurero que aparece en la columna de acciones.

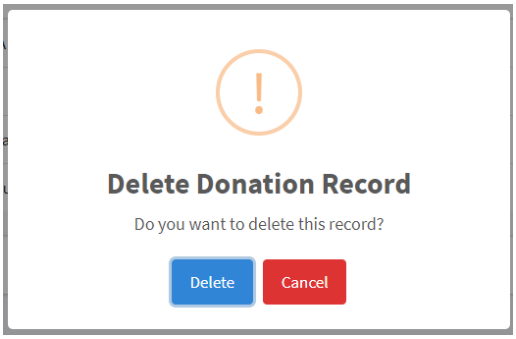


Figura 84 Ventana de eliminación de donación
Fuente: Elaboración propia

Reporte descargable de donaciones

En la figura 85 se muestra la ventana de confirmación a la hora de realizar la exportación del reporte descargable en Excel de donaciones. Para poder realizarlo se debe ingresar el rango de fechas a obtener la información y oprimir el botón de "Exportar Excel".

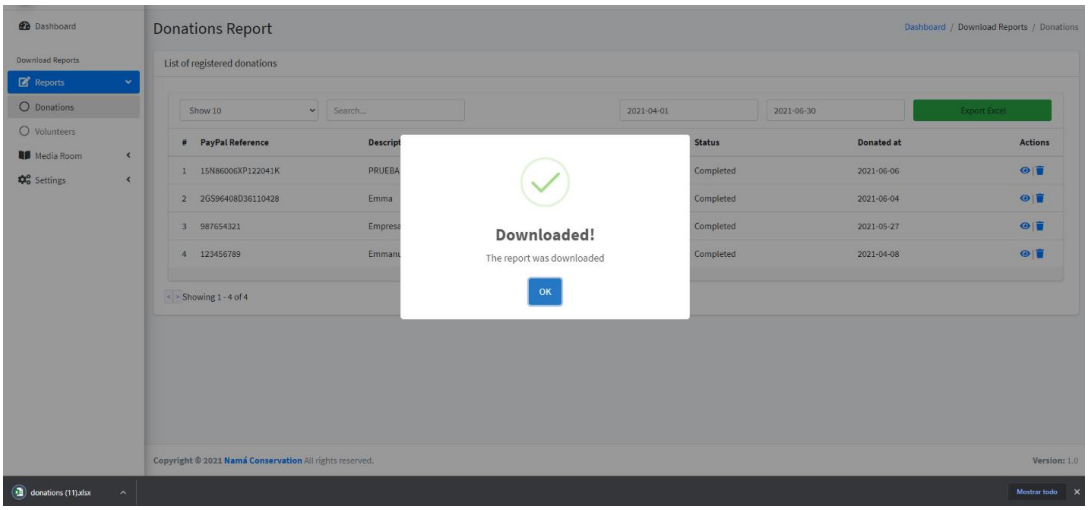


Figura 85 Confirmación de descarga de Excel de donaciones
Fuente: Elaboración propia

Formato del reporte descargable de donaciones

En la figura 86 se muestra el formato del reporte descargado tal y como se solicita en el requerimiento REQF-004.

	A	B	C	D	E	F
1	PayPal Reference	Description	Amount	Status	Comment	Donated At
2	123456789	Emmanuel	100	Completed	Hola gracias!	2021-04-08 00:00:00
3	987654321	Empresa Anonima	500	Completed	Buen trabajo	2021-05-27 00:00:00
4	2GS96408D36110428	Emma	100	Completed	Prueba local	2021-06-04 03:03:14
5	15N86006XP122041K	PRUEBA	100	Completed	MUY BUEN TRABAJO!	2021-06-06 00:18:30

Figura 86 Ejemplo de reporte exportable de donaciones
Fuente: Elaboración propia

5.7 PRUEBAS

En la etapa de pruebas de aceptación, se crea un subdominio de prueba llamado https://test.namaconservation.org/ el cual contiene el proyecto para poder realizar pruebas y para que el mismo sirva de presentación antes de publicarlo oficialmente.

A continuación, se describen las siguientes pruebas como resultado de cada una de las iteraciones realizadas, con el fin de garantizar el funcionamiento correcto de la aplicación web desarrollada.

5.7.1 Pruebas del módulo de donaciones

Prueba	Registro de donaciones
Actor	ACT-02
Requerimientos	REQF-001, REQF-002
Descripción	El ACT-02 debe ingresar a la página web, poder realizar la donación y que esta se registre en el sistema.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingresar a la página de donaciones	Mostrar formulario para la realización de donaciones	Se muestra el formulario de las donaciones	Aprobado
2	Ingresar la información correctamente	Mostrar la selección de método de pago	Se muestra la opción para realizar el pago por tarjeta de crédito/débito o cuenta de PayPal	Aprobado
3	No ingresar información en el espacio de nombre	Mostrar mensaje de error de espacio requerido	Se muestra un mensaje de color en rojo al no ingresar el nombre	Aprobado
4	Ingresar montos en las donaciones con cero, números negativos o letras.	No permitir que se puedan ingresar montos de cero, números negativos o letras	El espacio de monto de donaciones está bloqueado y solo admite números positivos mayor a uno	Aprobado

5	Mostrar los tipos de pago con cuenta PayPal o tarjeta de crédito	Mostrar las opciones pago disponibles	Se muestran las opciones para pago con tarjeta de crédito/débito o por cuenta PayPal	Aprobado
6	Realización de la donación	Se guarda la donación realizada en la base de datos	Se guardo la donación en la base de datos	Aprobado
7	Cancelación de donación	Retornar al formulario de donaciones	Se devuelve a la página donde está el formulario de donaciones	Aprobado
8	Visualización de mensaje de agradecimiento	Mostrar mensaje de agradecimiento tras la donación	Se muestra la pantalla con el mensaje de agradecimiento	Aprobado

Tabla 60 Pruebas de registro de donaciones
Fuente: Elaboración propia

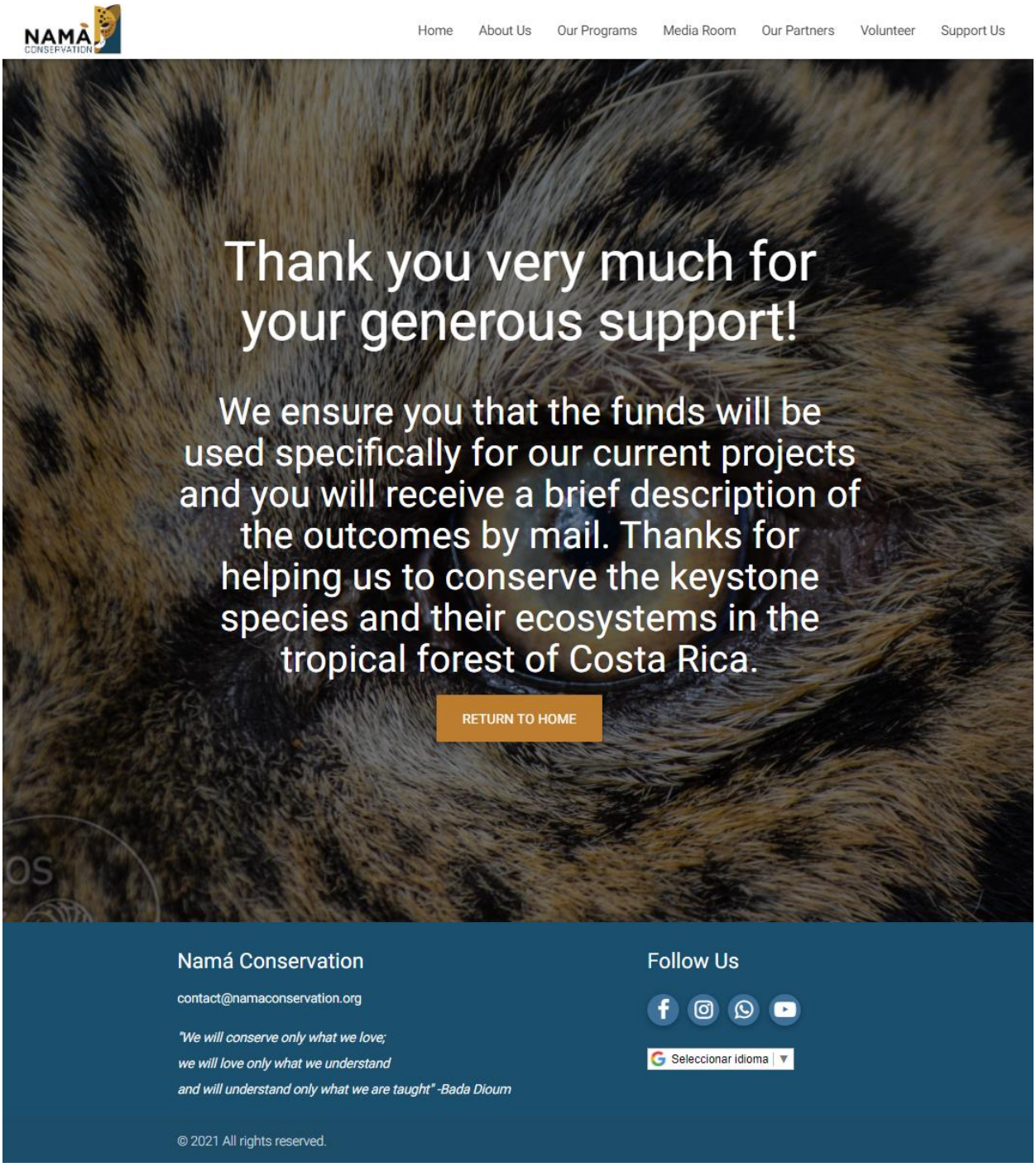


Figura 87 Prueba de registro de donación
Fuente: Elaboración propia

Prueba	Gestión de donaciones
Actor	ACT-01
Requerimientos	REQF-003, REQF-004,
Descripción	El ACT-01 debe poder ver las donaciones realizadas, buscarlas y descargarlas en un reporte de Excel según el rango de fechas.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingreso a la opción de donaciones	Visualizar la información de las donaciones realizadas	Se despliega una tabla con el contenido de las donaciones	Aprobado
2	Seleccionar un dato de la tabla desde el icono de vista	Se despliega la información con respecto a la donación	Se abre una ventana con toda la información de la donación	Aprobado
3	Ingresar texto en el espacio de búsqueda	Filtrar la tabla de donaciones con las coincidencias	Se muestran las donaciones que contienen coincidencias	Aprobado
4	Accionar botón de descarga de reporte	Se descarga un Excel según las fechas de rango ingresadas	Se descarga el Excel con las donaciones realizadas en el rango	Aprobado

Tabla 61 Pruebas de gestión de donaciones
Fuente: Elaboración propia

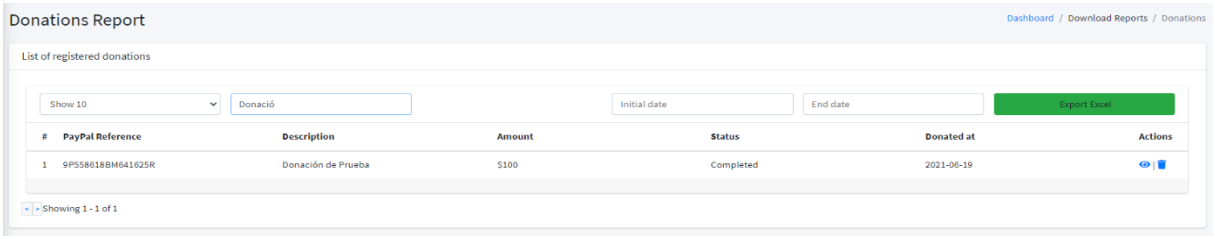


Figura 88 Prueba de tabla filtrada por búsqueda
Fuente: Elaboración propia

5.7.2 Pruebas del módulo de voluntariado

Prueba	Registro de voluntarios
Actor	ACT-02
Requerimientos	REQF-005
Descripción	El ACT-02 puede ingresar al formulario de voluntariado, llenar los datos y registrarse como voluntario para los programas que tiene el conservatorio.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingresar a la página de voluntariado	Mostrar formulario para la inscripción de voluntariado	Se muestra el formulario de voluntariado	Aprobado
2	Selección de programas del conservatorio.	Se debe mostrar y seleccionar diferentes programas	Se muestran y se seleccionan los diferentes programas	Aprobado
3	Selección de intereses.	Se debe mostrar y seleccionar diferentes intereses	Se muestran y se seleccionan los diferentes intereses	Aprobado
4	Ingresar la información con los espacios del formulario incompleto	Mostrar mensaje de error en espacios requeridos	Se muestra un mensaje de color en rojo debajo de cada espacio que este pendiente de ser llenado	Aprobado
5	Ingresar la información correctamente	Mostrar mensaje de confirmación del registro del voluntario	Se muestra el mensaje de confirmación de registro	Aprobado

Tabla 62 Pruebas de registro de voluntariados
Fuente: Elaboración propia

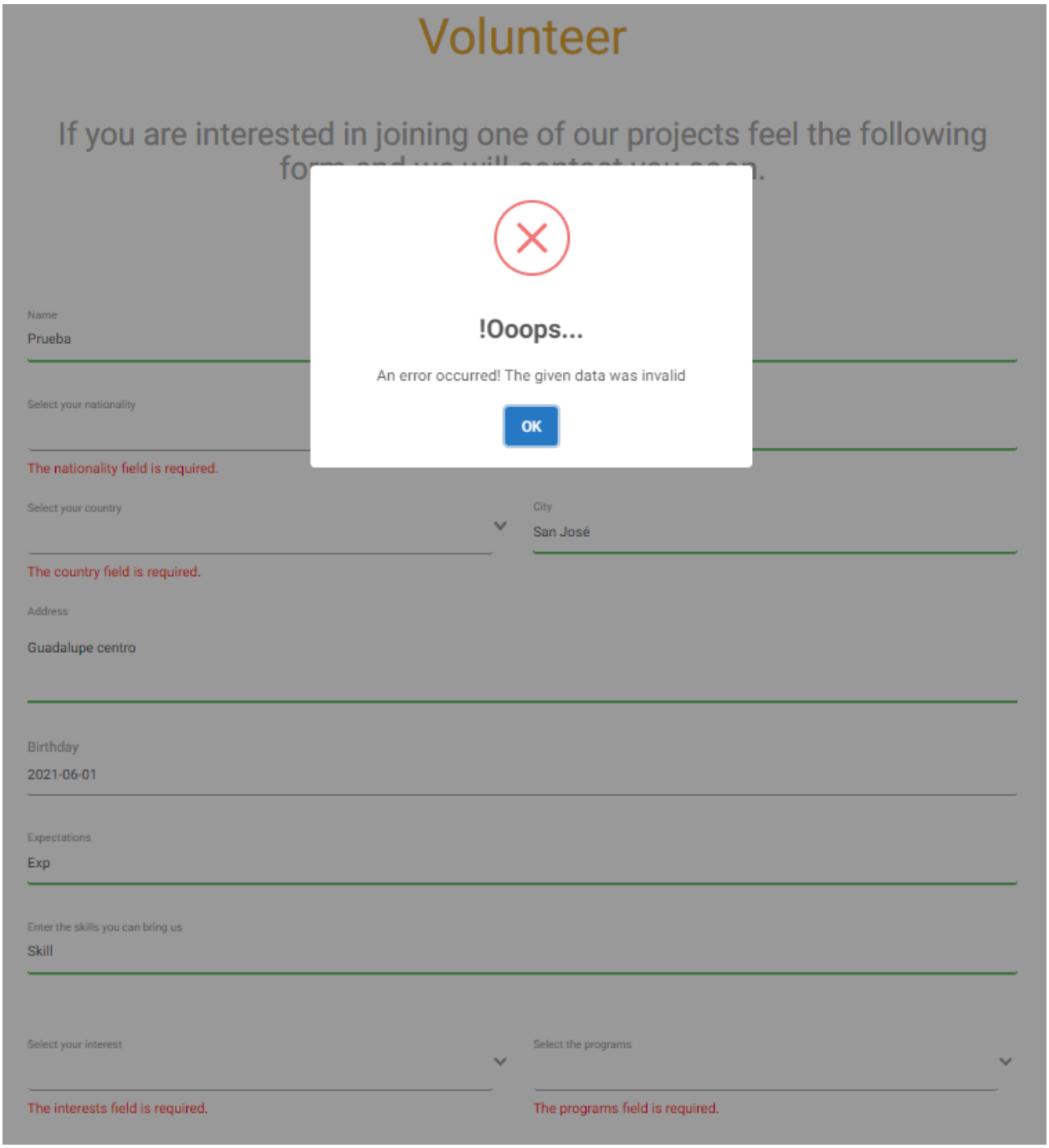


Figura 89 Prueba del formulario de voluntariado incompleto
Fuente: Elaboración propia

Prueba	Gestión de voluntariados
Actor	ACT-01
Requerimientos	REQF-006, REQF-007, REQF-008
Descripción	El ACT-01 debe poder ver los voluntarios registrados, buscarlos y ser descargados en un reporte de Excel según el rango de fechas.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingreso a la opción de voluntariado	Visualizar la información de los voluntarios registrados	Se despliega una tabla con los voluntarios registrados	Aprobado
2	Seleccionar un dato de la tabla desde el icono de vista	Se despliega la información del voluntario	Se abre una ventana con toda la información del voluntario	Aprobado
3	Ingresar texto en el espacio de búsqueda	Filtrar la tabla de voluntariado con las coincidencias	Se muestran los voluntarios que contienen coincidencias	Aprobado
4	Accionar botón de descarga de reporte	Se descarga un Excel con la información de los voluntarios según las fechas de rango	Se descarga correctamente el Excel con los voluntarios dentro del rango y se muestra un mensaje de descarga	Aprobado
5	Eliminar un voluntario de la tabla desde el icono de basurero	Se elimina el voluntario registrado y este no es mostrado en el sistema	Se elimina el voluntario y no se muestra en el sistema	Aprobado

Tabla 63 Pruebas de gestión de voluntariados
Fuente: Elaboración propia

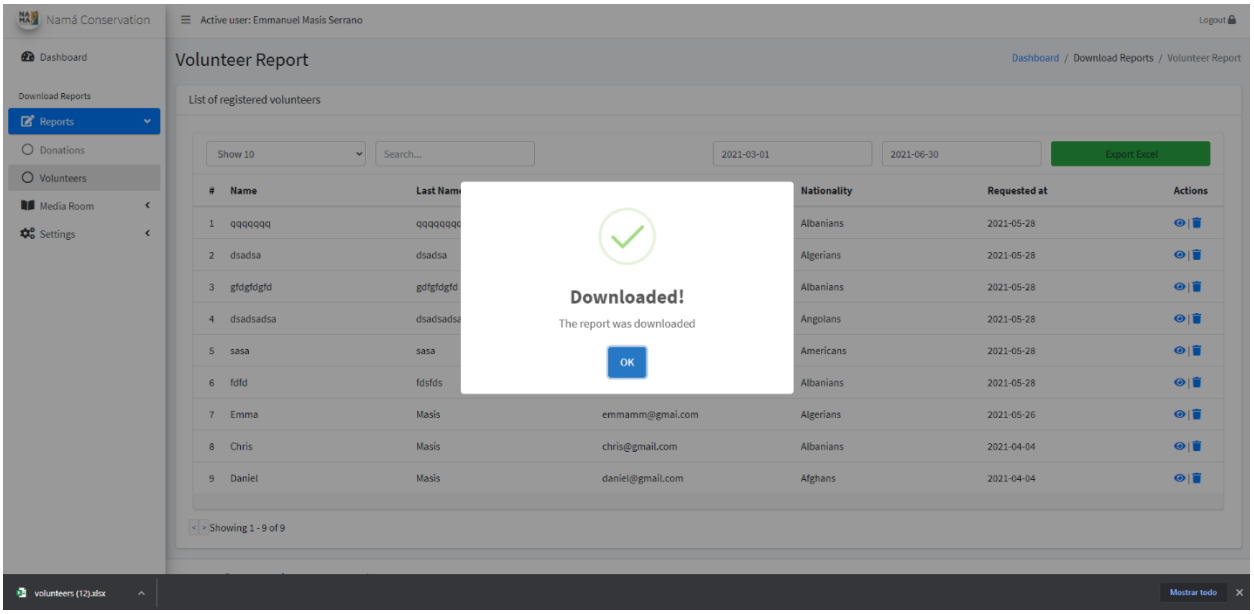


Figura 90 Prueba de descarga de Excel de voluntariado
Fuente: Elaboración propia

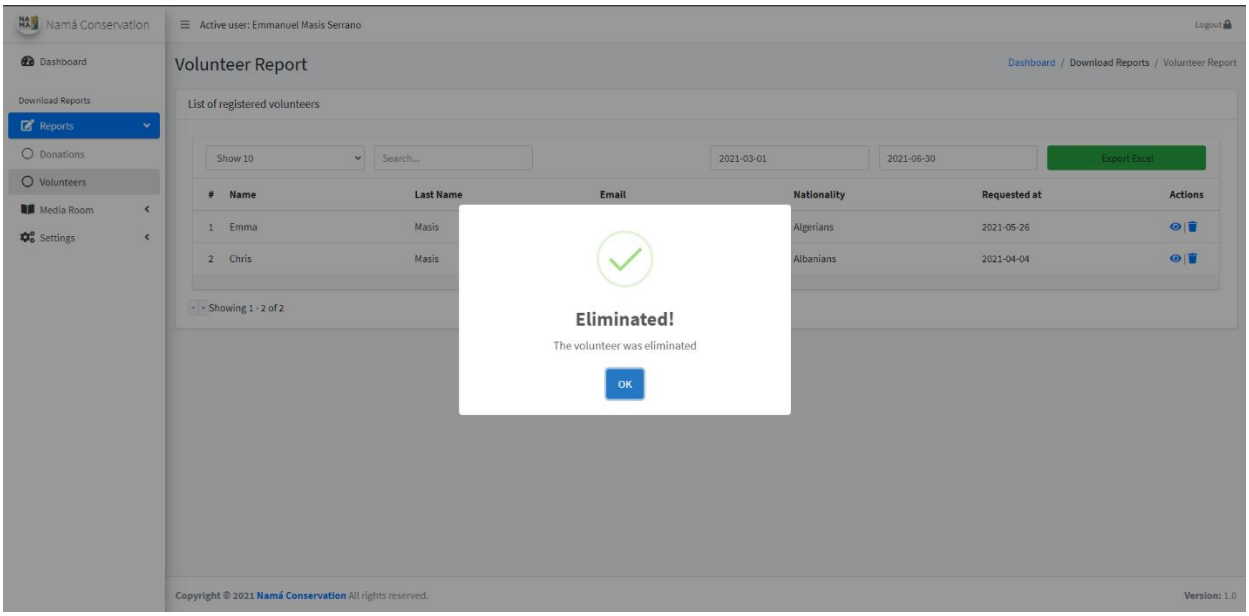


Figura 91 Prueba de eliminación de voluntario
Fuente: Elaboración propia

5.7.3 Pruebas del módulo de publicaciones

Prueba	Gestión de publicaciones
Actor	ACT-01
Requerimientos	REQF-009, REQF-010, REQF-011
Descripción	El ACT-01 puede crear, modificar, eliminar y visualizar publicaciones del conservatorio.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingreso a la opción de publicaciones	Visualizar las publicaciones realizadas	Se despliega una tabla con las publicaciones realizadas	Aprobado
2	Accionar el botón de crear publicación	Mostrar el formulario para la creación de publicaciones	Se abre una ventana con el formulario para crear la publicación	Aprobado
3	Accionar el botón de guardar la publicación	Guardar la publicación en la base de datos	La publicación es almacenada en la base de datos	Aprobado
4	Seleccionar un dato de la tabla desde el icono de editar	Mostrar la información de la publicación para modificarla	Se abre una ventana con la información de la publicación para poder realizarle cambios	Aprobado
5	Seleccionar un dato de la tabla desde el icono de eliminar	Mostrar mensaje para la eliminación de la publicación	Se muestra la ventana de confirmación para la eliminación	Aprobado

Tabla 64 Pruebas de gestión de publicaciones
Fuente: Elaboración propia

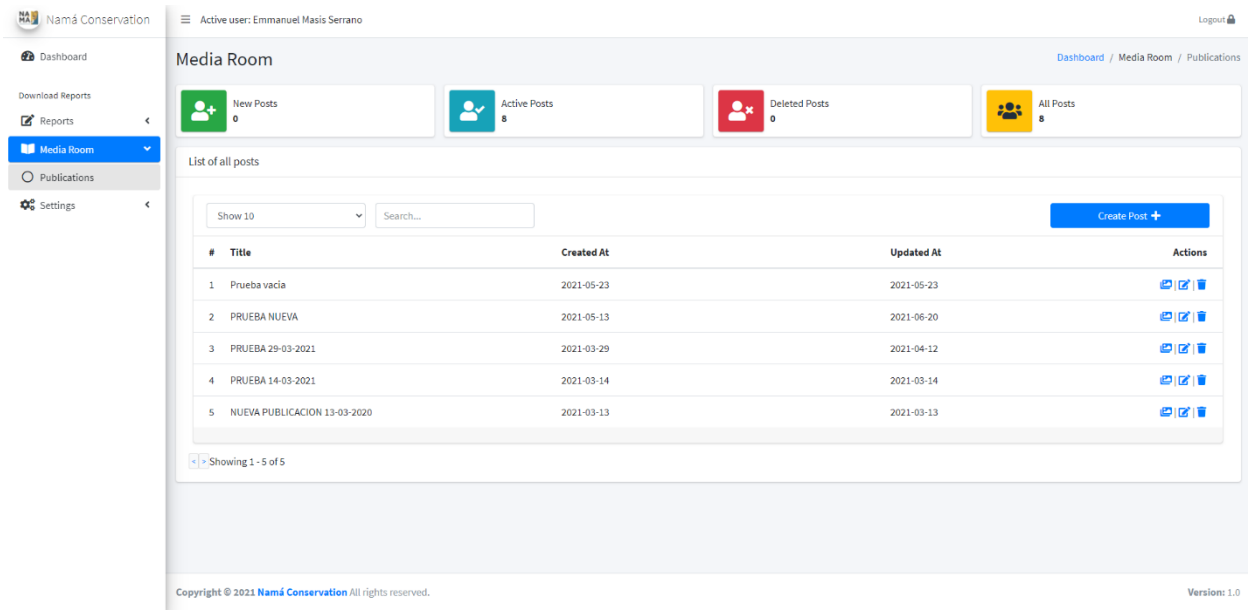


Figura 92 Prueba de visualización de publicaciones
Fuente: Elaboración propia

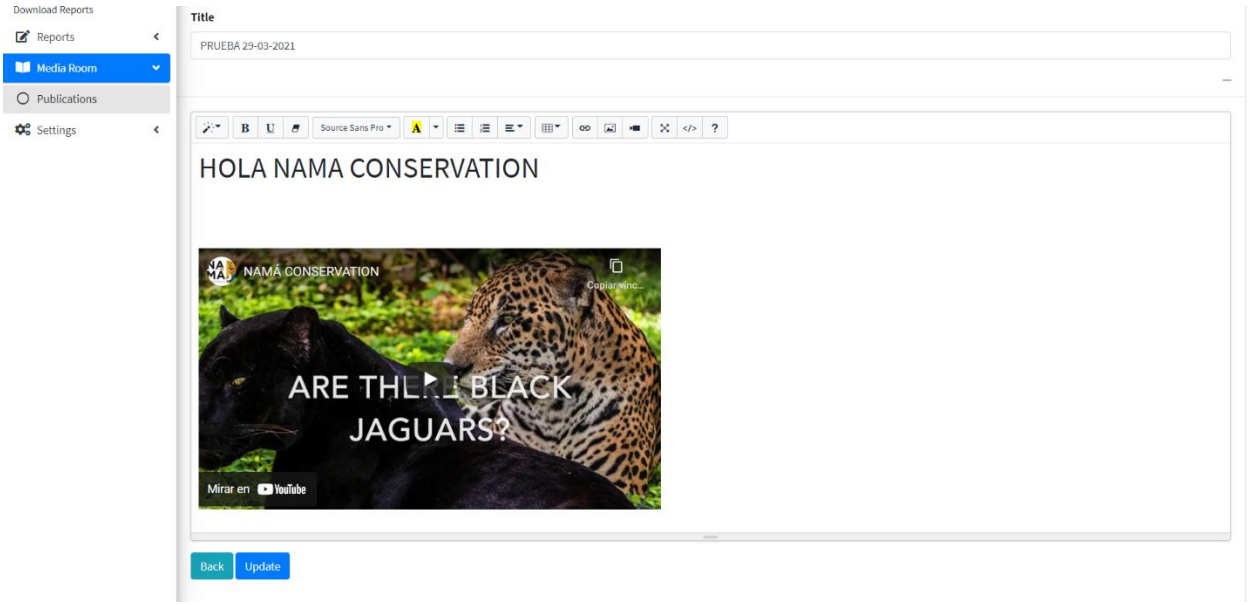


Figura 93 Prueba de visualización de publicación a editar
Fuente: Elaboración propia

5.7.4 Pruebas de la gestión de intereses

Prueba	Gestión de intereses
Actor	ACT-01
Requerimientos	REQF-014
Descripción	El ACT-01 puede crear, modificar, eliminar y visualizar los intereses de las personas que desean hacer voluntariado.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingreso a la opción de intereses	Visualizar los intereses existentes	Se despliega una tabla con los intereses del sistema	Aprobado
2	Accionar el botón de crear interés	Mostrar el campo de creación de interés	Se abre una ventana de dialogo para la creación del interés	Aprobado
3	Accionar el botón de guardar interés	Guardar el interés en la base de datos	El interés es guardado en la base de datos	Aprobado
4	Seleccionar un dato de la tabla desde el icono de editar	Mostrar el interés para modificarlo	Se abre una ventana de dialogo con el interés a modificar	Aprobado
5	Seleccionar un dato de la tabla desde el icono de eliminar	Mostrar mensaje para la eliminación del interés	Se muestra la ventana de confirmación para la eliminación del interés	Aprobado

Tabla 65 Pruebas de gestión de intereses
Fuente: Elaboración propia

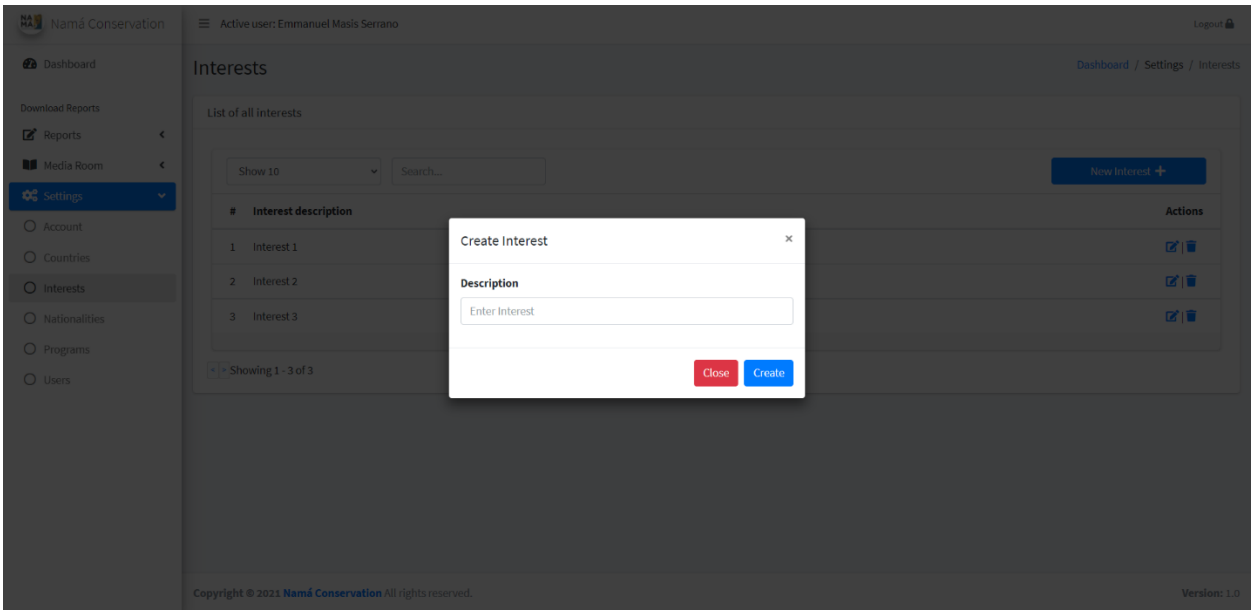


Figura 94 Prueba de mostrar campo de creación de interés
Fuente: Elaboración propia

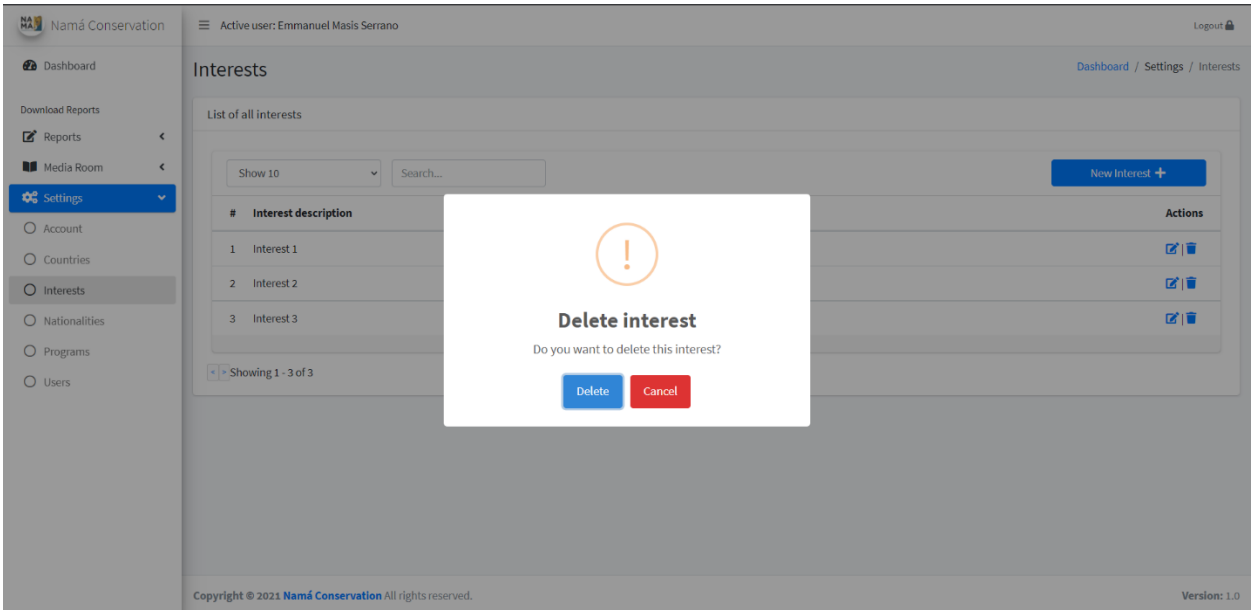


Figura 95 Prueba de mostrar mensaje de eliminación de interés
Fuente: Elaboración propia

5.7.5 Pruebas de la gestión de programas

Prueba	Gestión de programas
Actor	ACT-01
Requerimientos	REQF-015
Descripción	El ACT-01 puede crear, modificar, eliminar y visualizar los programas disponibles para realizar voluntariados.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingreso a la opción de programas	Visualizar los programas existentes	Se despliega una tabla con los programas del sistema	Aprobado
2	Accionar el botón de crear programa	Mostrar el campo de creación de programa	Se abre una ventana de dialogo para la creación del programa	Aprobado
3	Accionar el botón de guardar programa	Guardar el programa en la base de datos	El programa es guardado en el sistema	Aprobado
4	Seleccionar un dato de la tabla desde el icono de editar	Mostrar el programa para modificarlo	Se abre una ventana de dialogo con el programa a modificar	Aprobado
5	Seleccionar un dato de la tabla desde el icono de eliminar	Mostrar mensaje para la eliminación del programa	Se muestra la ventana de confirmación para la eliminación del programa	Aprobado

Tabla 66 Pruebas de gestión de programas
Fuente: Elaboración propia

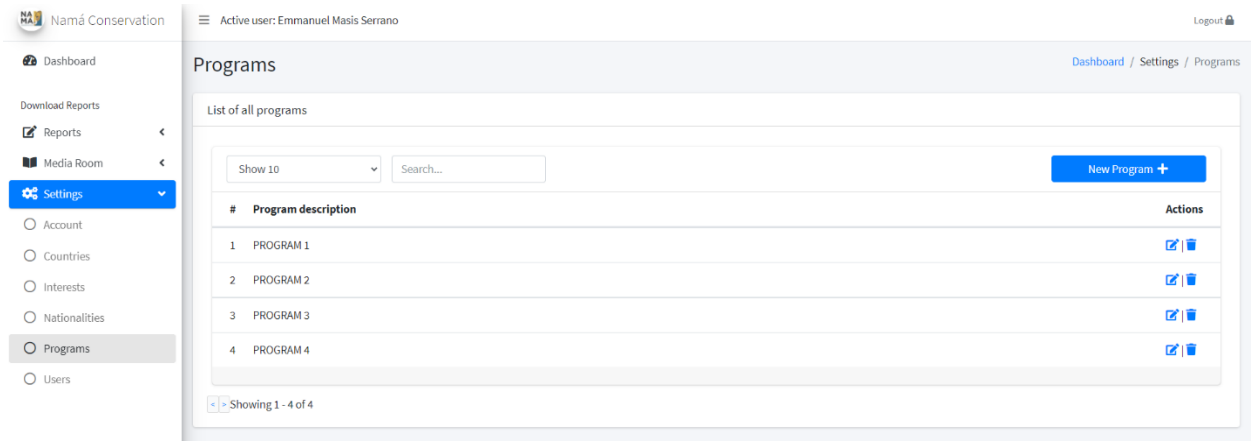


Figura 96 Prueba de visualización de programas
Fuente: Elaboración propia

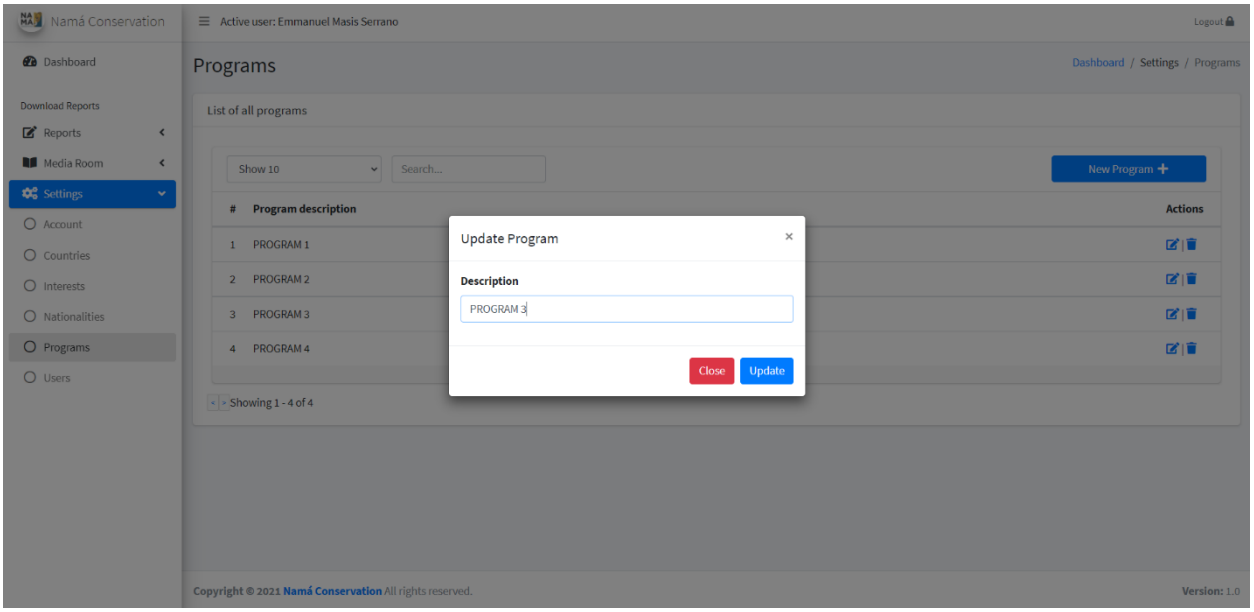


Figura 97 Prueba de visualización de programa a editar
Fuente: Elaboración propia

5.7.6 Pruebas de la gestión de usuarios

Prueba	Gestión de usuarios
Actor	ACT-01
Requerimientos	REQF-013
Descripción	El ACT-01 puede crear, modificar, eliminar y visualizar los usuarios una vez que ha sido autenticado por el sistema.

#	Acciones	Resultado esperado	Resultado actual	Estado
1	Ingreso de la página del login	Mostrar la pantalla de inicio de sesión	Se muestra la pantalla de login con los espacios de usuario y contraseña	Aprobado
2	Ingreso de datos incorrectos en el login	Mostrar mensaje de error al iniciar sesión	Se muestra un texto de error en los espacios donde son datos incorrectos	Aprobado
3	Accionar la opción de olvido de contraseña	Verificación de correo existente y envío de correo de restablecimiento	Se verifica que el correo existe en la base de datos y se reenvía correo con el link de restablecimiento de contraseña	Aprobado
4	Ingreso de datos correctos en el login	Permitir el acceso del usuario al sistema	El usuario tiene acceso a la pantalla principal del sistema	Aprobado

5	Ingreso a la opción de usuarios	Visualizar los usuarios existentes del sistema	Se despliega una tabla con los usuarios del sistema	Aprobado
6	Accionar el botón de crear usuario	Mostrar el formulario de creación de usuario	Se abre una ventana con el formulario para la creación del usuario	Aprobado
7	Ingreso de datos incompletos o incorrectos en la creación de usuario	Mostrar mensaje de error en los espacios requeridos	Se muestra un texto en rojo con el error en cada espacio incorrecto o incompleto	Aprobado
8	Accionar el botón de guardar usuario	Mostrar mensaje de confirmación y guardar el usuario en la base de datos	Se muestra el mensaje de confirmación y se guarda en la base de datos	Aprobado
9	Seleccionar un dato de la tabla desde el icono de editar	Mostrar el usuario para modificarlo	Se abre una ventana de con el usuario a modificar	Aprobado
10	Seleccionar un usuario de la tabla desde el icono de inactivar	Mostrar mensaje para la inactivación del usuario e inhabilitarlo del sistema	Se muestra la ventana de confirmación para la inactivación del usuario y se restringe el acceso de ese usuario en la aplicación	Aprobado

Tabla 67 Pruebas de gestión de usuarios
Fuente: Elaboración propia

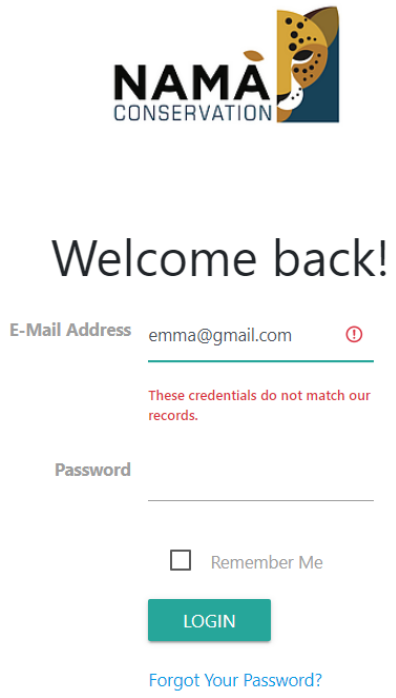


Figura 98 Prueba de ingreso de datos incorrectos en el login
Fuente: Elaboración propia

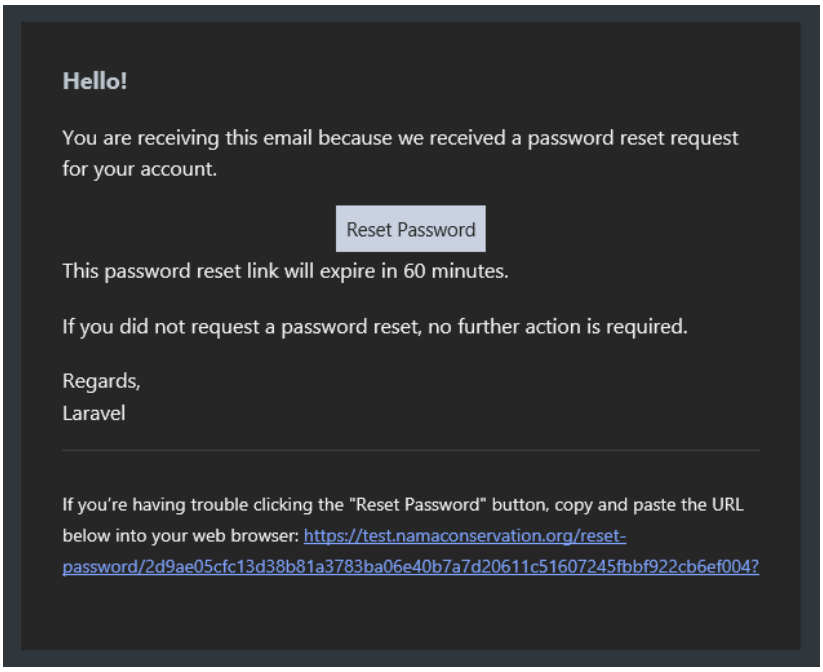


Figura 99 Prueba del correo con link de restablecimiento de contraseña
Fuente: Elaboración propia

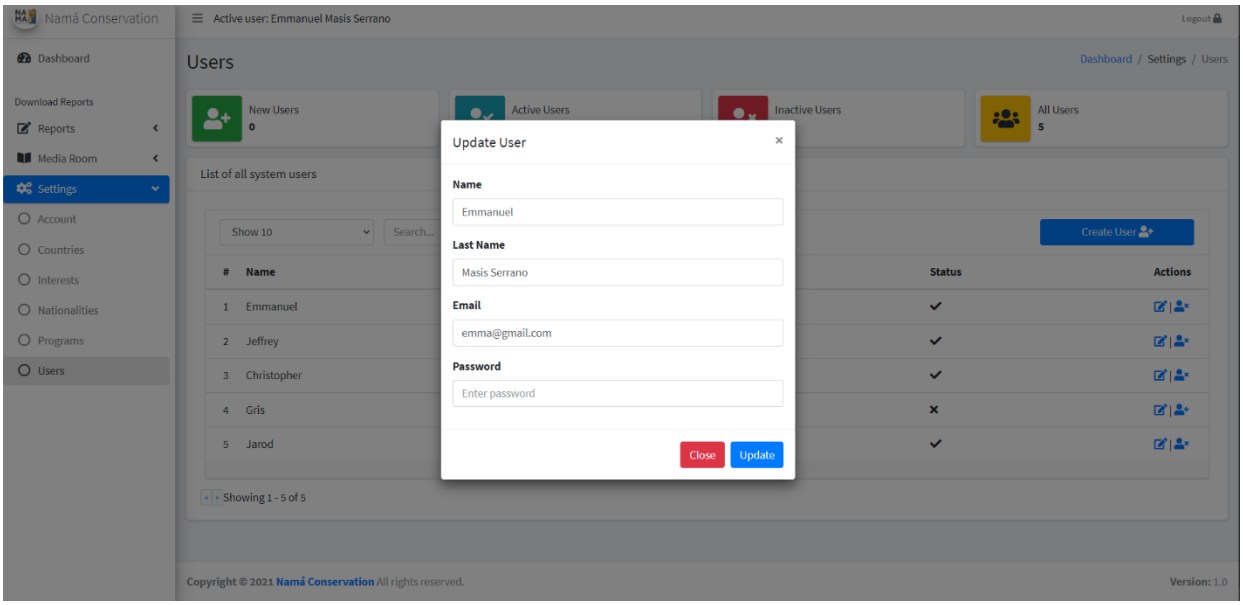


Figura 100 Prueba de mostrar usuario para editarlo
Fuente: Elaboración propia

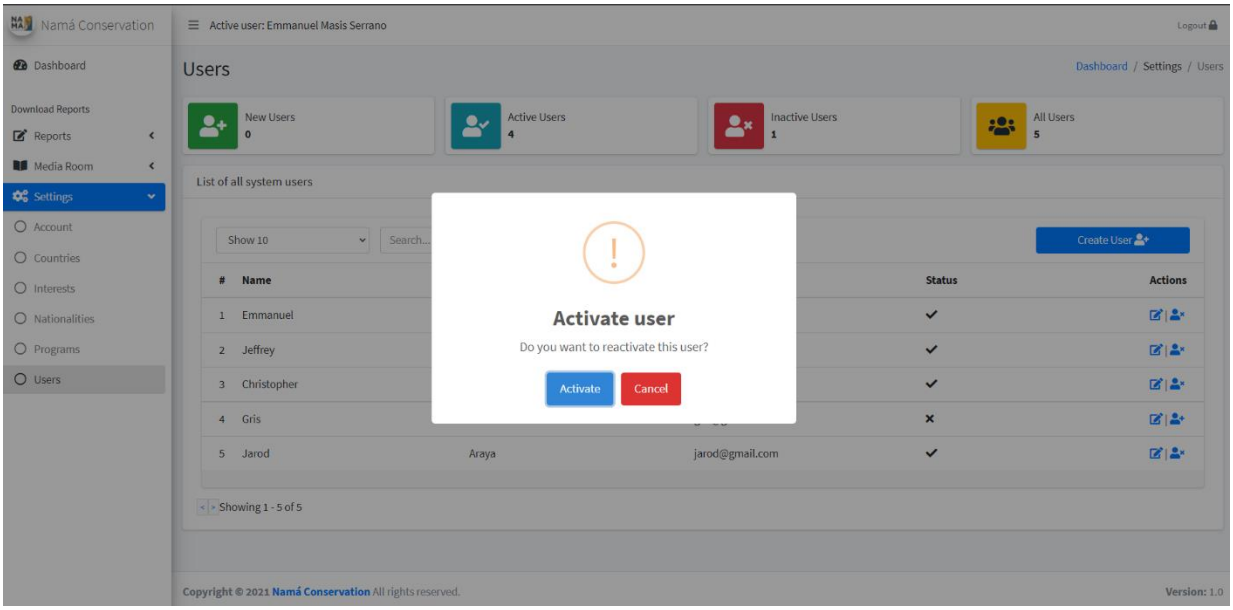


Figura 101 Prueba de inactivación de usuario
Fuente: Elaboración propia

5.8 IMPLEMENTACIÓN

Una vez que son realizadas las pruebas de aceptación satisfactoriamente, se procede a la carga del proyecto en el servidor, para ello se realizan los siguientes pasos:

- Se accede al panel de control del servicio de hosting y se selecciona la opción de "Administrador de archivos" como lo muestra la figura 102.

Figura 102 Panel de control del hosting
Fuente: Hostgator

- Dentro de la opción de Administrador de archivos, se selecciona la carpeta llamada public_html que es la carpeta donde el servidor almacena cualquier información con respecto a sitios web. Una vez posicionado dentro de la carpeta se selecciona la opción de "subir archivos" para cargar el proyecto al servidor como lo muestra la figura 103.

Figura 103 Carga de proyecto
Fuente: Hostgator

- Una vez cargado el proyecto en el servidor, se procede a crear el subdominio https://official.namaconservation.org/ donde en el espacio de “raíz de documento” se ingresa la dirección public_html/nama_conservation/public. La carpeta nama_conservation contiene el proyecto realizado y la carpeta public contiene la conexión de la aplicación de Laravel con el servidor como lo muestra la figura 104.

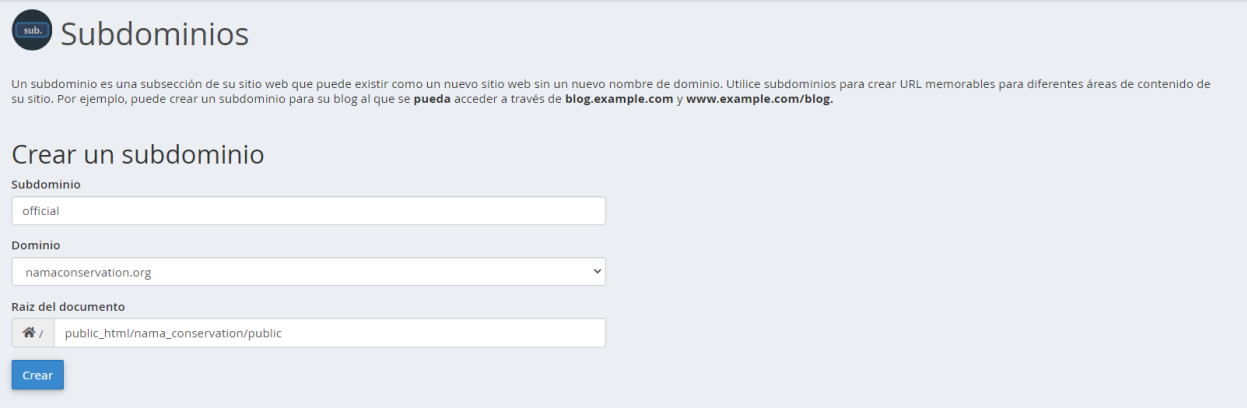


Figura 104 Creación de subdominio
Fuente: Hostgator

- Con el proyecto cargado y el subdominio creado, se procede a crear la base de datos en el servidor junto con el usuario de base de datos para el acceso de la información de la aplicación.

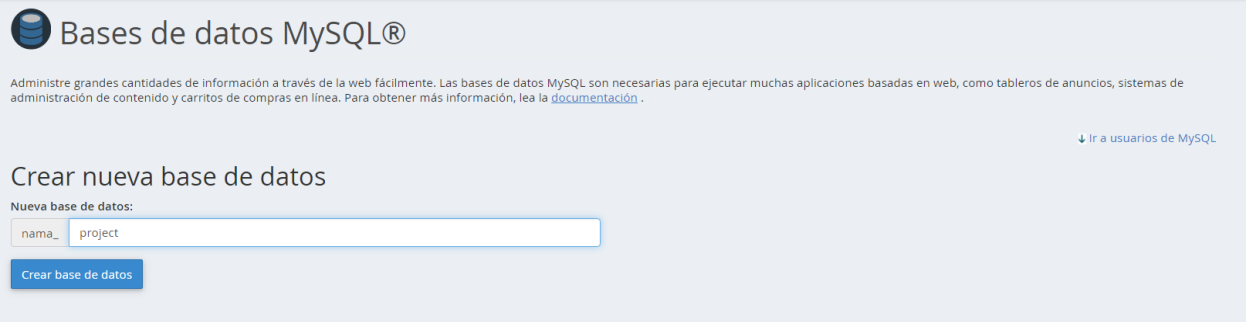


Figura 105 Creación de base de datos
Fuente: Hostgator

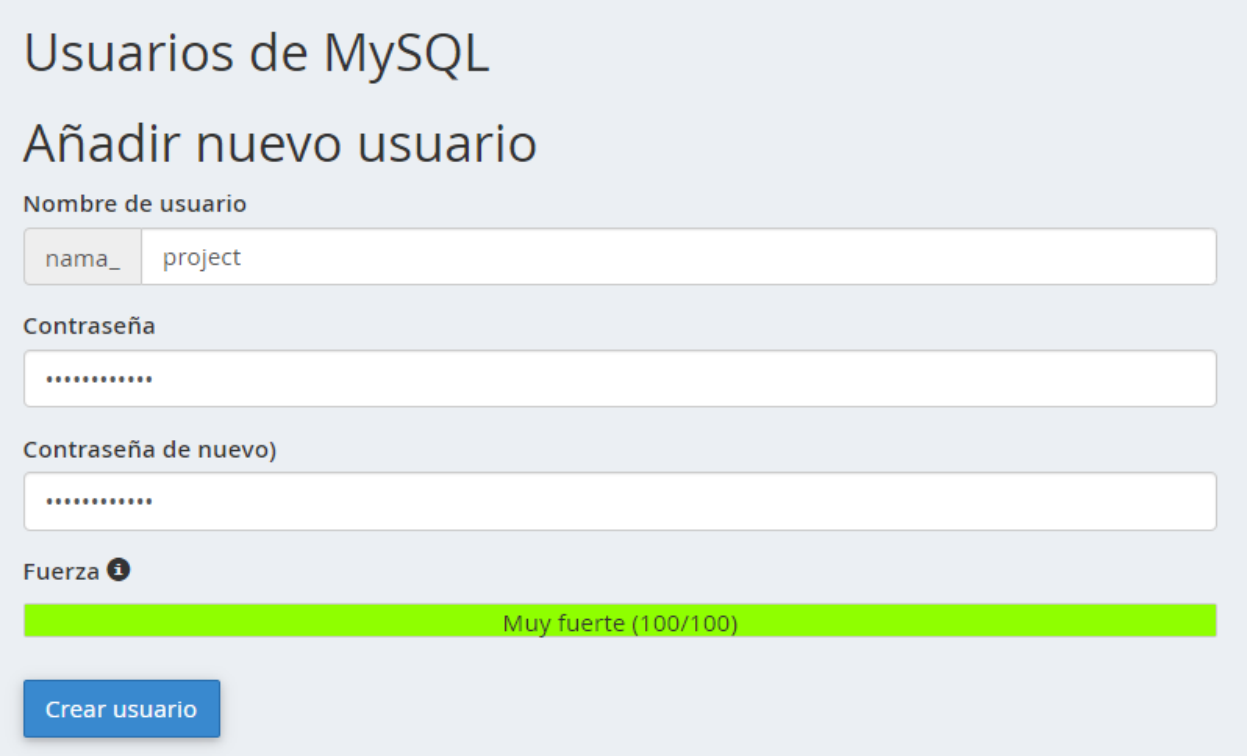


Figura 106 Creación de usuario de base de datos
Fuente: Hostgator

- Luego se asigna el usuario creado a la base de datos con el nombre de nama_project.

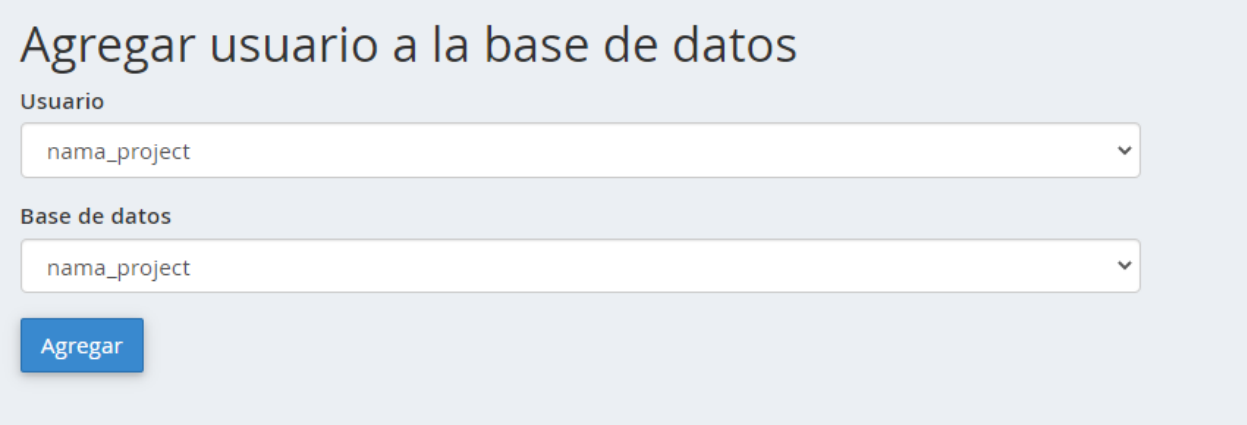


Figura 107 Asignación de usuario a base de datos
Fuente: Hostgator

- Por último, se ingresa a la página https://official.namaconservation.org/ para poder verificar que la aplicación web esté funcionando correctamente y se verifica que el certificado SSL esté realizando la conexión segura. Ver figura 108 y 109.

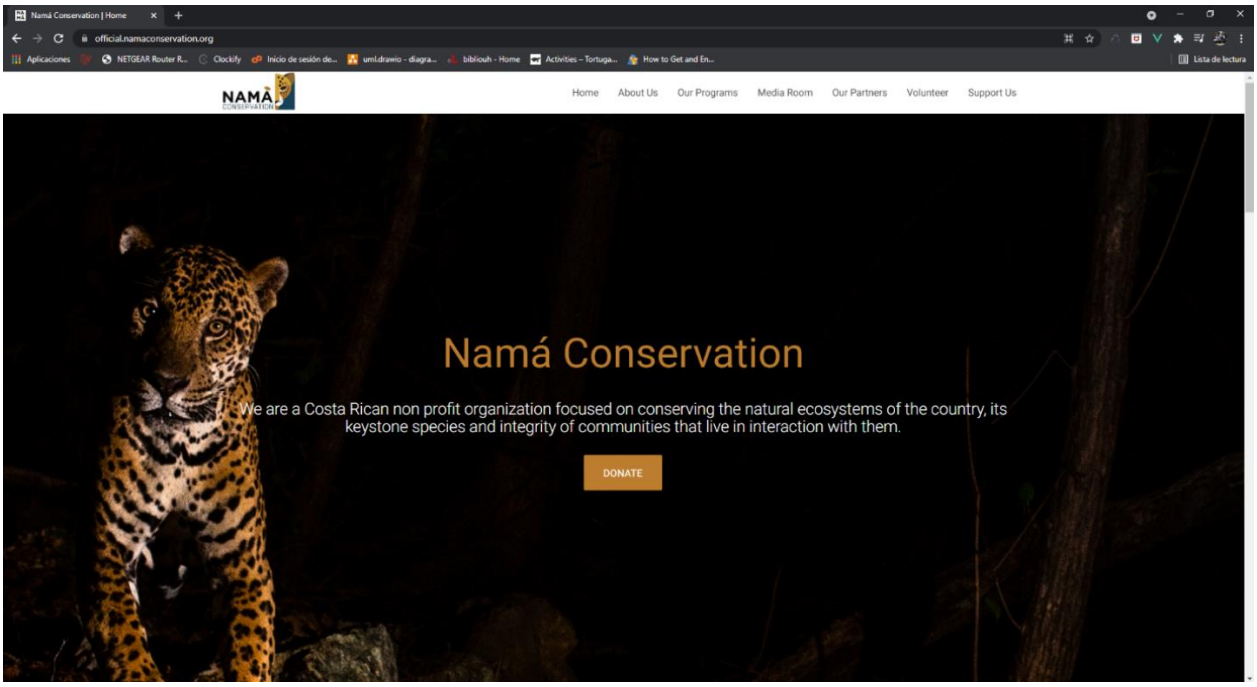


Figura 108 Página web implementada
Fuente: Elaboración propia

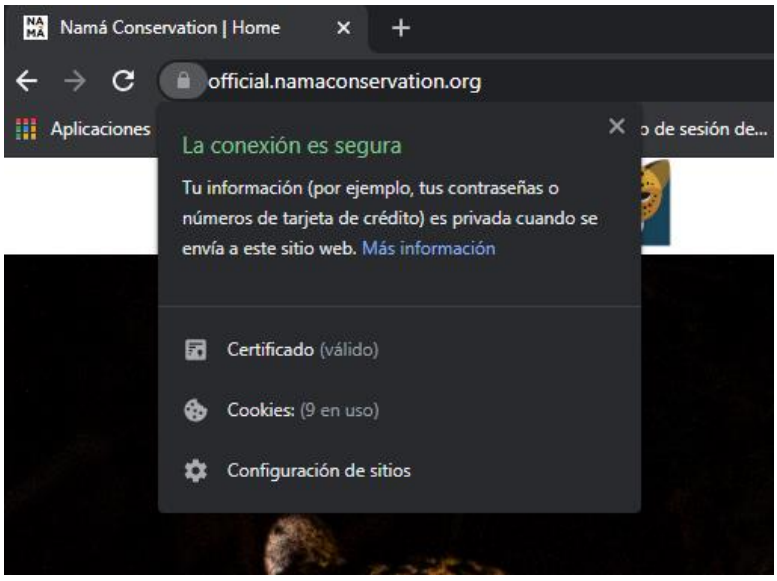


Figura 109 Certificado SSL
Fuente: Elaboración propia

CAPÍTULO VI: CONCLUSIONES Y RECOMENDACIONES

En este capítulo se describen las conclusiones y recomendaciones que fueron obtenidas a lo largo del desarrollo del proyecto, con las cuales se pretende brindar un resumen detallado del resultado final de la aplicación web para la ONG Namá Conservation.

6.1 Conclusiones

La aplicación web desarrollada cumple con las necesidades de la ONG, porque facilita, agiliza y mejora el procedimiento para la recaudación de donaciones monetarias, logrando que los internautas puedan realizar una donación a la ONG sin intermediación o contacto por parte del personal de Namá Conservation que retrase la efectuación de la donación por la realización del contacto, por otro lado, facilita al internauta la realización de la donación porque el mismo podrá efectuarla por medio de tarjeta de crédito/débito o por la misma cuenta de PayPal.

Además, con la aplicación web, Namá Conservation podrá llevar una base de datos con las personas que desean ser parte de los programas de voluntariados, para que a la hora de realizar una campaña puedan seleccionar y reclutar a las personas que residen cerca de los diferentes proyectos que tienen, contemplando los programas e intereses de las actividades que quieren realizar las diferentes personas.

Y por último con el módulo de las publicaciones, Namá Conservation podrá tener un medio oficial para realizar sus publicaciones con respecto a los proyectos realizados en los diferentes parques nacionales de Costa Rica, donde podrán mostrar a la comunidad de internet la forma en la cual ellos invierten el dinero recaudado de las donaciones.

Para el primer objetivo específico “Analizar e identificar el proceso actual de la organización con respecto a la gestión para recibir donaciones y solicitudes de voluntariados, mediante técnicas o herramientas de recolección de datos” se realizó una entrevista al equipo de Namá Conservation para poder entender cuáles eran los procedimientos para la recaudación de donaciones y reclutamiento de personas para los programas de voluntariado.

Una vez obtenida la información sobre la ONG y generado el diagnóstico de la situación actual se determina que la organización necesitaba realizar una reestructuración en sus procesos,

además de conseguir una innovación tecnológica que les ayudara en sus procedimientos para eficientizar las recaudaciones de donaciones y el reclutamiento de las personas para los voluntariados.

El segundo objetivo definido como “Identificar los requerimientos funcionales y no funcionales junto con las necesidades que se deben contemplar en el proyecto” para este objetivo se realizó el levantamiento de requerimientos como se muestra en la sección 5.2 donde como resultado, se establece las necesidades funcionales que la ONG tiene con respecto a la implementación de la aplicación web y los aspectos técnicos con los que debe de contar para que el proyecto sea exitoso.

Para el tercer objetivo “Diseñar los diagramas de base de datos y los diagramas de casos de uso, del nuevo proceso de recolección de donativos y solicitudes de voluntariados que permita cubrir las necesidades plasmadas en los requerimientos mediante lenguaje UML” una vez que son realizados los diagramas según los requerimientos obtenidos, se determina que el proceso para la recaudación de donaciones es simplificado, logrando acortar el tiempo para la realización de la transacción monetaria sin intermediación del personal de Namá Conservation. Por otro lado, para las solicitudes de voluntariados se define el módulo donde va a estar concentrada la información para la selección de personas registradas para los voluntariados.

Para el cuarto objetivo definido como “Desarrollar el sistema web para mejorar el proceso de recaudación de donaciones, publicaciones y solicitudes de voluntariados bajo el patrón de arquitectura MVC” se desarrolló la aplicación web utilizando el framework de Laravel bajo el modelo de estructura MVC con el uso de JavaScript y PHP para darle un funcionamiento dinámico a la aplicación. Por lo anterior, se establece una nueva herramienta tecnológica con la que Namá Conservation contará para optimizar sus procesos actuales, logrando traer a la organización una innovación tecnológica.

Para el último objetivo “Implementar la aplicación web desarrollada en el hosting, con el fin de poner en marcha la solución del problema principal” se concluye que la aplicación web implementada será el medio oficial por el que Namá Conservation dará a conocer su organización, demostrando que las donaciones monetarias ingresadas están destinadas en sus

diferentes proyectos e invitando siempre a los cibernautas a continuar donando para el mantenimiento del conservatorio.

6.2 Recomendaciones

De acuerdo con el desarrollo del proyecto y las conclusiones definidas anteriormente, se recomienda como medida general, realizar revisiones periódicas de mantenimiento para verificar que el funcionamiento de la aplicación web sea el correcto, ya que con el tiempo muchas tecnologías nuevas van a ir apareciendo o actualizaciones de navegadores web, bases de datos, PHP, JavaScript, PayPal entre otros, que pueden ocasionar un desperfecto o deterioro en el funcionamiento correcto de la aplicación web. Para ello se recomienda dar mantenimiento a la aplicación web una vez puesta en el servidor la primera versión del proyecto.

Además, se brindan las siguientes recomendaciones:

- El proyecto no contemplo la realización de un manual de usuario, por lo tanto, se recomienda realizar una documentación con respecto al nuevo proceso para la recaudación de donaciones, registro, selección y reclutamiento de los voluntariados y la gestión para la creación de publicaciones, para que exista un respaldo de los nuevos procesos en caso de que alguna persona deje de laborar para la ONG.

- En caso de que se ocupe realizar ajustes, mantenimientos o levantamiento de nuevos requerimientos, se recomienda que se continúe con el framework en MVC de Laravel junto con el lenguaje de programación presente en este proyecto, ya que la escalabilidad de Laravel junto con su sistema de versionado de migraciones en base de datos, se ajustan para nuevos desarrollos, modificaciones, etc.

- A nivel de base de datos, se recomienda realizar una tarea con el servicio hosting para que cada cierto tiempo se pueda crear un respaldo de la información dentro de un rango de tiempo prudente.

- Se recomienda el almacenamiento del código desarrollado en algún repositorio o servicio de la nube, para que se cuente con un respaldo en caso de que exista algún problema con el servicio de hosting o por algún ataque cibernético.

- Para la aplicación web publicada, se recomienda calendarizar las fechas de los pagos del servicio de hosting, ya que se requiere de una anualidad para el mantenimiento del dominio, certificados SSL, correos u otros. Estos pueden dejar de funcionar por la falta de la anualidad y en el caso de la página web dejará de ser accesible para los cibernautas una vez vencida la anualidad.

BIBLIOGRAFÍA

Alonso, R. (2020). ¿Qué es un hosting o alojamiento web?. Recuperado de https://miposicionamientoweb.es/que-es-un-hosting/#que-es-hosting-que-significa

Aranda Vera, Á. (2016). Instalación y parametrización del software (UF1893). Málaga, España: IC Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/44522

B. G. (2020). Los mejores frameworks PHP para desarrolladores web. Obtenido de https://www.hostinger.es/tutoriales/mejores-frameworks-php#2-CodeIgniter

B. G. (2019). Qué es jQuery. Obtenido de https://www.hostinger.es/tutoriales/que-es-jquery

Baena Paz, G. (2014). Metodología de la investigación. México D.F, México: Grupo Editorial Patria. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/40362

Blasco, F. (2019). Programación orientada a objetos en Java. Bogotá, Ediciones de la U. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/127125

Capterra. (s.f.). Obtenido de https://www.capterra.es/software/48791/postgresql

Carvajal Palomares, F. (2017). Instalación y configuración del software de servidor Web: UF1271. Madrid, España: Editorial CEP, S.L. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh

Casado Iglesias, C. (2015). Entornos de desarrollo. Madrid, España: RA-MA Editorial. Recuperado de http://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/62495

Casas Roma, J. y i Caralt, J. C. (2014). Diseño conceptual de bases de datos en UML. Barcelona, España: Editorial UOC. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/57635

Ceballos Sierra, J. (2015). Visual Basic.NET: lenguaje y aplicaciones (3a. ed.). Madrid, España: RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/59726

Cíceri Vásquez, J. (2018). Introducción a Laravel (1ª. ed.). Buenos Aires, Argentina, Six Ediciones.

Cuatrecasas Arbós, L. (2012). Procesos en flujo Pull y gestión Lean: sistema Kanban. Madrid, España: Ediciones Díaz de Santos. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/62616

del Castillo, C. C. y Olivares Orozco, S. (2014). Metodología de la investigación. México D.F, México: Grupo Editorial Patria. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/39410

Dorantes, C. A. (2015) Laravel, el mejor framework en PHP. Obtenido de https://platzi.com/blog/laravel-framework-php/

Eslava Muñoz, V. J. (2018). El nuevo PHP: conceptos avanzados. Madrid, España: Bubok Publishing S.L. Recuperado de http://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/51353

Fresno Chávez, C. (2019). Metodología de la investigación: así de fácil. Córdoba, El Cid Editor. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/98278

García Rubio, F. O. Piattini Velthuis, M. G. y García Rodríguez de Guzmán, I. (2018). Calidad de sistemas de información (4a. ed.). Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106509

García Rubio, F. O. Piattini Velthuis, M. G. y García Rodríguez de Guzmán, I. (2019). Calidad de Sistemas de Información (4a. ed.). Bogotá, Ediciones de la U. Recupe0rado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/127061

Garzás Parra, J. Piattini Velthuis M. G. (2015) Fabricas de Software: Experiencias, Tecnologías y Organización (2ª. ed.). Madrid, RA-MA Editorial.

Genero Bocco, M. Cruz Lemus, J. A. y Piattini Velthuis, M. G. (2014). Métodos de investigación en ingeniería del software. Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106450

Guerrero Dávila, G. (2015). Metodología de la investigación. México D.F, México: Grupo Editorial Patria. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/40363

Guillermina María Eugenia Baena Paz (2014). Metodología de la investigación. México D.F, México: Grupo Editorial Patria. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/40362

Laravel. (s.f.). Queues. Obtenido de Laravel: https://laravel.com/docs/master/queues

López Mendoza, M. (2020). Extreme Programming: Qué es y cómo aplicarlo. Obtenido de https://openwebinars.net/blog/extreme-programming-que-es-y-como-aplicarlo/

López Quijado, J. (2014). Domine PHP y MySQL (2a. ed.). Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106410

López Sanz, M. (2015). Programación web en el entorno servidor. Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106487

López Sanz, M. (2015). Programación web en el entorno servidor. Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106487

Lucidchart. (s.f.). Qué es un diagrama de flujo. Recuperado de https://www.lucidchart.com/pages/es/que-es-un-diagrama-de-flujo/#section_1

Mailify. (s.f.). ¿Qué es un dominio y un subdominio? Recuperado de https://www.mailify.com/es/help/entregabilidad/nombres-de-dominios/que-es-un-dominio

Maldonado, M. (2018). Qué es Ruby on Rails: usos en el desarrollo de software. Obtenido de https://www.digital55.com/desarrollo-tecnologia/ruby-on-rails-desarrollo-software/

Martínez Ruiz, H. (2012). Metodología de la investigación. México, D.F, México: Cengage Learning. Recuperado de https://elibro.net.uh.remotexs.xyz/es/lc/bibliouh/titulos/39957.

Molina Rios, J. R. (2019). “SWIRL” la metodología para el diseño y desarrollo de aplicaciones web (1ª. ed.). Alicante, España: Área de innovación y desarrollo, SL.

Monroy Mejía, M. D. L. Á. y Nava Sánchez, N. (2018). Metodología de la investigación. Grupo Editorial Éxodo. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/172512

Monte Galiano, J. (2016). Implantar scrum con éxito. Barcelona, España: Editorial UOC. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/58575

Otálora-Luna, J. E. Callejas-Cuervo, M. y Alarcón-Aldana, A. C. (2018). Metamodelo de medición de esfuerzo en proyectos de desarrollo de software. Tunja, Editorial UPTC. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/132832

Otálora-Luna, J. E. Callejas-Cuervo, M. y Alarcón-Aldana, A. C. (2018). Metamodelo de medición de esfuerzo en proyectos de desarrollo de software. Tunja, Editorial UPTC. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/132832

Oviedo Regino, E. (2015). Lógica de programación orientada a objetos. Bogotá, Colombia: Ecoe Ediciones. Recuperado de http://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/70431

Parada, M. (2019). Qué es SQL Server. Obtenido de https://openwebinars.net/blog/que-es-sql-server/

Pavón Puertas, J. (2014). Creación de un portal con PHP y MySQL (4a. ed.). Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106413

Pérez Carvajal, R. J. (2016). Mantenimiento del software (UF1894). Málaga, España: IC Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/44523

Pérez Rodríguez, M. D. (Coord.) (2013). Programación de páginas web dinámicas con Apache, Base de Datos MySQL y PHP (2a. ed.). Editorial ICB. Recuperado de http://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/113229

Pérez, L. Pérez, R. y Seca, M. V. (2020). Metodología de la investigación científica. Editorial Maipue. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/138497

Piattini Velthuis, M. Vizcaíno Barceló, A. y García Rubio, F. O. (2014). Desarrollo global de software. Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106438

Piattini Velthuis, M. y García Rubio, F. (2015). Calidad de sistemas de información (3a. ed.). RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106390

Pulido Romero, E. Escobar Domínguez, Ó. y Núñez Pérez, J. Á. (2019). Base de datos. Grupo Editorial Patria. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/121283

Rad, N y Turley, F. (2019). Los fundamentos de agile Scrum (1ª. ed.). Hertogenbosch, Editorial Van Haren Publishing B.V.

Recio García, J. A. (2016). HTML5, CSS3 y JQuery: curso práctico. Paracuellos de Jarama, Madrid, RA-MA Editorial. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/106494

Rondón, C. (2016). Tipos de Middleware en Laravel. https://styde.net/tipos-de-middleware-en-laravel/

Sarasa, A. (2016). Introducción a las bases de datos NoSQL usando MongoDB. Barcelona, España: Editorial UOC. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/58524

Solís, J. (2014). ¿Qué es Bootstrap y cómo funciona en el diseño web? . Obtenido de: https://www.arweb.com/blog/%C2%BFque-es-bootstrap-y-como-funciona-en-el-diseno-web/

Sommerville, I. (2011). Ingeniería de software (9a. ed.). México, D.F, México: Pearson Educación. Recuperado de http://elibro.net.uh.remotexs.xyz/es/lc/bibliouh/titulos/37857

Sommerville, I. (2011). Ingeniería de software (9a. ed.). Pearson Educación. https://elibro.net.uh.remotexs.xyz/es/lc/bibliouh/titulos/37857

Teniente López, E. Costal Costa, D. y Sancho Samsó, M. R. (2015). Especificación de sistemas software en UML. Barcelona, España: Universitat Politècnica de Catalunya. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/6140

Terán Anciano, J. (2016). Manual de Introducción al lenguaje HTML. Formación para el Empleo. Madrid, España: Editorial CEP, S.L. Recuperado de https://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/50964

Vara Mesa, J. M. Verde Marín, J. y López Sanz, M. (2015). Desarrollo web en entorno servidor. Madrid, España: RA-MA Editorial. Recuperado de http://elibro.net.uh.remotexs.xyz/es/ereader/bibliouh/62489

Verisign. (s.f.). Todo lo que debe de saber sobre certificados SSL. Recuperado de https://www.verisign.com/es_LA/website-presence/online/ssl-certificates/index.xhtml#:~:text=SSL%20(Secure%20Sockets%20Layer%20o,navegador%20y%20un%20servidor%20web.&text=Autenticar%20la%20identidad%20del%20sitio,est%C3%A1n%20en%20un%20sitio%20falso.

ANEXOS

Anexo A. Carta de anuencia de realización de proyecto

Namá Conservation
Heredia, Costa Rica

T: +506.8396-6812

Ciudad de Heredia, 18 de enero de 2021

Señores

Universidad Hispanoamericana

Por medio de la presente hago constar que el estudiante Edwin Emmanuel Masís Serrano, portador de la cédula de identidad 1-1296-0995, se encuentra realizando el proyecto "Desarrollo e implementación de aplicación web para la ONG Namá Conservation", ubicado en Condominio Quinta Fontana, San Pablo, Heredia.

Sin otro particular se extiende la presente a solicitud del interesado el día 18 de Enero de 2021.

Atentamente,

M.Sc. Juan Carlos Cruz
Namá Conservation
carloscruz@namaconservation.org

Anexo B. Carta de aceptación de realización del proyecto

Namá Conservation
Heredia, Costa Rica

T: +506.8396-6812

Ciudad de Heredia, 21 de Mayo de 2021

Señores

Universidad Hispanoamericana

Por medio de la presente hago constar que el estudiante Edwin Emmanuel Masís Serrano, portador de la cédula de identidad 1-1296-0995, comenzó su proyecto "Desarrollo e implementación de aplicación web para la ONG Namá Conservation", ubicado en Condominio Quinta Fontana, San Pablo, Heredia desde el 18 de enero de 2021.'

El señor Emmanuel desarrolló aplicación web con el contenido provisto por nuestra organización, donde permite la realización de donaciones monetarias y registro de voluntariados, el cuál hizo entrega el 21 de mayo de 2021, mismo que funcionó de manera óptima después de realizar las pruebas respectivas, además de cumplir con todos los requerimientos y entregables en el tiempo establecido. Dicho producto será de gran utilidad para nuestros esfuerzos de conservación, así como para la divulgación de la información que se genera constantemente por nuestra organización en pro de la conservación en Costa Rica.

Quisiera manifestar nuestro agradecimiento con la calidad del trabajo desempeñado, así como el compromiso y la buena disposición del estudiante que mostró durante todo el desarrollo del proyecto.

Sin otro particular, se extiende la presente a solicitud del interesado el día 21 de Mayo de 2021.

Atentamente,

M.Sc. Juan Carlos Cruz
Namá Conservation
carloscruz@namaconservation.org

Anexo C. Cuestionario de donaciones y voluntariado en Google Forms

Namá Conservation

Sign Up

*Obligatorio

Quiero ayuda como/ I want to help like:

- ◯ Voluntario/ Volunteer
- ◯ Donador/Donor

Nombre/ Name *

Tu respuesta

Cédula/ ID Number *

Tu respuesta

Teléfono de contacto/ Phone number *

Tu respuesta

Email *

Tu respuesta

Enviar