



SOLICITUD DE TRÁMITE DE CARTAS DE TRABAJO DE GRADUACIÓN DE EPS

FACULTAD DE INGENIERIA
UNIDAD DE EPS

Nombre del estudiante:: Juan Diego Alvarado Salguero

CUI: 2998050930101 Registro Académico: 201807335

Carrera: Ingeniería en Ciencias y Sistemas Teléfono: 57099747

Email: 2998050930101@ingenieria.usac.edu.gt

Fecha de incorporación al programa de EPS: 14/02/2024



Fecha de finalización del programa de EPS: 24/02/2025

Fecha de realización de la evaluación final de EPS: 25/04/2025

Nombre de los examinadores:

- 1) Inga. Floriza Avila
- 2) Ing. Carlos Azurdia
- 3) Ing. Sergio Leonel Gómez Bravo

Nombre del(a) Asesor(a)-Supervisor(a) de EPS: Ing. Luis Fernando Cajas

Firma del(a) Asesor(a)-Supervisor(a) de EPS:  Luis Fernando Cajas Calijau
Ingeniero en Ciencias y Sistemas
Colegiado No. 17,648 Firma del estudiante 

Vo.Bo. Coordinador de Área de EPS: _____

ADJUNTO

- 1. Informe final completo
- 2. 1 CD con el contenido del informe final
- 3. Ficha de seguimiento de trabajo de graduación de EPS (impresa en ambas caras de la hoja)
- 4. Finiquito original extendido por la empresa/institución donde se realizó el EPS, (solo para Ingeniería Civil y para EPS de tres meses como sustituto de trabajo de graduación)
- 5. Si el asesor es externo a la Unidad de EPS, carta del asesor, firmada y sellada



Universidad de San Carlos de Guatemala
Facultad de Ingeniería
(Deberá adjuntarse el trabajo en todo proceso de corrección)

FICHA DE SEGUIMIENTO DE TRABAJOS DE GRADUACIÓN DE E.P.S



I. DATOS GENERALES:

1. Nombre del estudiante: JUAN DIEGO ALVARADO SALGUERO
 2. CUI: 2998050930101
 3. Registro Académico : 201807335
 4. Dirección: CIUDAD SANCRISTOBAL Z8 DE MIXCO
 5. Escuela: CIENCIAS Y SISTEMAS Carrera: INGENIERIA EN CIENCIAS Y SISTEMAS
- Correo electrónico: 2998050930101@ingenieria.usac.edu.gt

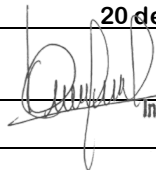
II. APROBACIÓN DE PROTOCOLO:

6. Fecha de Ingreso: 20 de febrero de 2024
7. Observaciones del revisor (a) de protocolos _____
8. Fecha de aprobación del protocolo: 20 de marzo de 2024
9. Título aprobado para el trabajo de graduación: SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO EN JEPA GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)
10. Nombre del asesor (a) aprobado (a): Ing. Luis Fernando Cajas
11. Vo.Bo. del coordinador (a) del área: _____

III. REVISIÓN DEL ASESOR (A) – SUPERVISOR (A) DE EPS.

12. Fecha de inicio del trabajo: 20 de marzo de 2024
13. Observaciones del asesor (a) / supervisor (a): _____
14. Fecha de aprobación del asesor (a)/ supervisor (a): 24 de febrero 2025

IV. REVISIÓN DEL ASESOR (A) DEL TRABAJO

15. Fecha de inicio del trabajo: 20 de marzo de 2024
16. Observaciones del asesor (a):

Luis Fernando Cajas Calijau
Ingeniero en Ciencias y Sistemas
Colegiado No. 17,648
17. Fecha de aprobación de asesor (a): 24 de febrero 2025

V. REVISIÓN DEL DIRECTOR DE LA UNIDAD DE E.P.S.

16. Fecha de revisión: _____

Vo.Bo. _____
Firma y sello

VI. REVISIÓN DE LA ESCUELA:

17. Fecha de revisión: _____

Vo.Bo. _____
Firma y sello

VII. REVISIÓN DE LINGÜÍSTICA:

18. Fecha de revisión: _____

19. Fecha de entrega del trabajo revisado: _____

20. Fecha de verificación de correcciones _____

f) _____
Firma y sello

VIII. REVISIÓN DEL DIRECTOR (A) DE ESCUELA:

21. Fecha de revisión del director (a) de escuela: _____

f) _____
Firma y sello

IX. REVISIÓN DEL SEÑOR DECANO

22. Fecha de entrega: _____
Firma y sello

X. ENTREGA DEL ORIGINAL IMPRESO A OFICINA DE LINGÜÍSTICA
Entrega de un original impreso para comprobación del proceso.

23. Fecha de entrega: _____ Vo.Bo. _____
Firma y sello

FIRMA DE CONFORMIDAD

f) 
Luis Fernando Cajas Calijau
Ingeniero en Ciencias y Sistemas
Colegiado No. 17,648
FIRMA Y SELLO DEL ASESOR (A)

f) 
ESTUDIANTE

NOTAS:

Este seguimiento es interno y no exime la redacción de las caras que para el efecto determine el Reglamento de Trabajos de Graduación de la Facultad de Ingeniería. Guatemala, junio 2000. Aprobado por Junta Directiva mediante acta No. 04-2007, Punto Noveno inciso 9.1, del 19 de febrero de 2007



Guatemala, 30 de abril de 2025

Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio hago de su conocimiento que mi persona, en representación de la Asociación Jepa Guatemala, brindó la asesoría necesaria para las modificaciones y finalización del informe final que detalla lo realizado en el Ejercicio Profesional Supervisado (EPS) del estudiante Juan Diego Alvarado, perteneciente a la carrera de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, identificado con el CUI 2998050930101 y registro académico 201807335, titulado "SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S, DESARROLLADO E IMPLEMENTADO EN JEPa GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS DE DESARROLLO (ASOCOPRODE ONG)". Así mismo, certifico haber revisado y validado la redacción final del informe para garantizar su calidad y pertinencia.

Nombre: **SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO EN JEPa GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)**, para Asociación Jepa Guatemala.

Sin otro particular, suscribo.

Atentamente,

Jordy Gabriel
Noriega
Morán

Firmado digitalmente
por Jordy Gabriel
Noriega Morán
Fecha: 2025.05.13
14:52:44 -06'00'

Lic. Jordy Gabriel Noriega Morán
Presidente Asociación Jepa Guatemala

c.c. JepaGt



Guatemala, 30 de abril de 2025

Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio, deseo informarles que el estudiante Juan Diego Alvarado, perteneciente a la carrera de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, identificado con el CUI 2998050930101 y registro académico 201807335, ha concluido de manera satisfactoria todas las actividades relacionadas con el desarrollo del proyecto denominado "**SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO EN JEPa GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)**". Este proyecto dio inicio el 14 de febrero de 2024, ha cumplido con todos los objetivos planteados y las horas correspondientes al Ejercicio Profesional Supervisado (EPS) en nuestra institución, culminando el 16 de agosto de 2024.

En virtud de lo expuesto anteriormente, extendiendo el presente FINIQUITO. Agradeciendo su colaboración y apoyo en la Asociación Jepa Guatemala. Sin otro particular me suscribo.

Jordy Gabriel Noriega Morán Firmado digitalmente
por Jordy Gabriel
Noriega Morán
Fecha: 2025.05.13
14:53:11 -06'00'

Lic. Jordy Gabriel Noriega Morán
Presidente Asociación Jepa Guatemala

c.c. JepaGt



Guatemala, 30 de abril de 2025

Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio hago de su conocimiento que el estudiante Juan Diego Alvarado, perteneciente a la carrera de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, identificado con el CUI 2998050930101 y registro académico 201807335, ha concluido satisfactoriamente el proyecto de Ejercicio Profesional Supervisado titulado **“SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG’S DESARROLLADOE IMPLEMENTADO EN JEPa GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)”**. De esta manera extendiendo mi aprobación y ratifico que el proyecto ha sido finalizado y cumple con los objetivos planteados.

Jordy
Gabriel
Noriega
Morán

Firmado
digitalmente por
Jordy Gabriel
Noriega Morán
Fecha: 2025.05.13
14:53:25 -06'00'

Lic. Jordy Gabriel Noriega Morán
Presidente Asociación Jepa Guatemala

c.c. JepaGt



Guatemala, 30 de abril de 2025

Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio le informo que después de revisar los avances del trabajo de EPS titulado **"SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO EN JEPA GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)"**, el cual está a cargo del estudiante de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, JUAN DIEGO ALVARADO, que se identifica con el registro académico 201807335 y CUI 2998050930101, hago constar que dicho proyecto se completó en su totalidad exitosamente, entregando todas las funcionalidades solicitadas y la documentación necesaria para su uso.

Agradeciendo la atención a la presente y quedando a sus órdenes para cualquier información adicional.

Atentamente,



Luis Fernando Cajas Calijau
Ingeniero en Ciencias y Sistemas
Colegiado No. 17,648

Ing. Luis Fernando Cajas
Profesor Interno
Asesor de trabajo de graduación
Colegiado No.17,648

Guatemala, 30 de abril de 2025


Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio le informo que después de revisar los avances del trabajo de EPS titulado **“SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG’S DESARROLLADO E IMPLEMENTADO EN JEPG GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)”** y JEPG Guatemala, el cual está a cargo del estudiante de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, JUAN DIEGO ALVARADO, que se identifica con el registro académico 201807335 y CUI 2998050930101, hago constar que el trabajo escrito ha sido revisado en su totalidad por mi persona, autorizando su publicación sin ningún inconveniente..

Agradeciendo la atención a la presente y quedando a sus órdenes para cualquier información adicional.

Atentamente,


Luis Fernando Cajas Calijau
Ingeniero en Ciencias y Sistemas
Colegiado No. 17,648

Ing. Luis Fernando Cajas
Profesor Interno
Asesor de trabajo de graduación
Colegiado No.17,648



Guatemala, 30 de abril de 2025

Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio hago de su conocimiento que mi persona, en representación de la Asociación Comunitaria de Proyectos de Desarrollo (ASOCOPRODE ONG), brindó la asesoría necesaria para las modificaciones y finalización del informe final que detalla lo realizado en el Ejercicio Profesional Supervisado (EPS) del estudiante Juan Diego Alvarado, perteneciente a la carrera de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, identificado con el CUI 2998050930101 y registro académico 201807335, titulado "SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S, DESARROLLADO E IMPLEMENTADO EN JEPG GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS DE DESARROLLO (ASOCOPRODE ONG)". Así mismo, certifico haber revisado y validado la redacción final del informe para garantizar su calidad y pertinencia.

Nombre: **SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO EN JEPG GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)**,
Asociación Comunitaria de Proyectos de Desarrollo (ASOCOPRODE ONG).
Sin otro particular, suscribo.
Atentamente,

Josué Misael Chacon Barahona
Presidente Asociación Comunitaria de Proyectos de Desarrollo (ASOCOPRODE ONG)



Guatemala, 30 de abril de 2025

Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio, deseo informarles que el estudiante Juan Diego Alvarado, perteneciente a la carrera de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, identificado con el CUI 2998050930101 y registro académico 201807335, ha concluido de manera satisfactoria todas las actividades relacionadas con el desarrollo del proyecto denominado **SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO EN JEPA GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)**. Este proyecto dio inicio el 14 de febrero de 2024, ha cumplido con todos los objetivos planteados y las horas correspondientes al Ejercicio Profesional Supervisado (EPS) en nuestra institución, culminando el 16 de agosto de 2024.

En virtud de lo expuesto anteriormente, extendiendo el presente FINIQUITO. Agradeciendo su colaboración y apoyo en la Asociación Comunitaria de Proyectos de Desarrollo (ASOCOPRODE ONG). Sin otro particular me suscribo-

Josué Misael Chacon Barahona
Presidente Asociación Comunitaria de Proyectos de Desarrollo (ASOCOPRODE ONG)



Guatemala, 30 de abril de 2025

Ingeniero Oscar Argueta Hernández
Director Unidad de Ejercicio de Práctica Supervisado -EPS
Facultad de Ingeniería.

Respetable Ingeniero Argueta:

Por este medio hago de su conocimiento que el estudiante Juan Diego Alvarado, perteneciente a la carrera de Ingeniería en Ciencias y Sistemas de la Universidad de San Carlos de Guatemala, identificado con el CUI 2998050930101 y registro académico 201807335, ha concluido satisfactoriamente el proyecto de Ejercicio Profesional Supervisado titulado **“SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG’S DESARROLLADO E IMPLEMENTADO EN JEPA GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)”**. De esta manera extendiendo mi aprobación y ratifico que el proyecto ha sido finalizado y cumple con los objetivos planteados.

Josué Misael Chacon Barahona
Presidente Asociación Comunitaria de Proyectos de Desarrollo (ASOCOPRODE ONG)



Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería en Ciencias y Sistemas

**SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO
EN JEPG GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)**

Juan Diego Alvarado Salguero

Asesorado por Ing. Luis Fernando Cajas Calijau

Guatemala, febrero de 2025

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA



FACULTAD DE INGENIERÍA

**SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO
EN JEPG GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)**

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA

POR

JUAN DIEGO ALVARADO SALGUERO

ASESORADO POR ING. LUIS FERNANDO CAJAS CALIJAU

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO EN CIENCIAS Y SISTEMAS

GUATEMALA, FEBRERO DE 2025

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA



NÓMINA DE JUNTA DIRECTIVA

DECANO A.I.	Ing. José Francisco Gómez Rivera
VOCAL II	Ing. Mario Renato Escobedo Martínez
VOCAL III	Ing. José Milton de León Bran
VOCAL IV	Ing. Kevin Vladimir Cruz Lorente
VOCAL V	Ing. Francisco José Paz González
SECRETARIA	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. José Francisco Gómez Rivera (a. i.)
EXAMINADOR(A)	Inga. Floriza Avila
EXAMINADOR(A)	Ing. Carlos Azurdia
EXAMINADOR(A)	Ing. Sergio Leonel Gómez Bravo
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

SISTEMA DE GESTIÓN DE INFORMACIÓN DE ONG'S DESARROLLADO E IMPLEMENTADO EN JEPA GUATEMALA Y ASOCIACIÓN COMUNITARIA DE PROYECTOS (ASOCOPRODE)

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería en Ciencias y Sistemas, con fecha de agosto del año 2024.

A handwritten signature in black ink, appearing to read 'Juan Diego', with a horizontal line drawn through it.

Juan Diego Alvarado Salguero

ACTO QUE DEDICO A:

Dios

A Dios por darme la sabiduría necesaria para poder afrontar todos los retos que conllevo estudiar Ingeniería y sobre todo porque jamás se sintió una falta de su amor y misericordia en mi vida

Mis padres

Edwin Alvarado y Nora Salguero por su amor, comprensión, sus consejos y sobre todo por siempre mostrarme su apoyo incondicional en cada momento de mi vida.

Mi hermano

Hugo Alvarado, por siempre estar conmigo, por apoyarme en los buenos y malos momentos. Por siempre ser un hermano resiliente y comprensivo hacia mi persona.

Mis primas

Nathalie, Mónica consideradas también como mis hermanas les agradezco siempre por el cariño brindado y sobre todo por siempre apoyarme y aconsejarme a darlo todo por mi carrera y a cumplir mis sueños.

Mis amigos

En general, a todos mis amigos con los que he compartido experiencias y he aprendido a ser mejor persona enfrentando cualquier adversidad, infinitas gracias.

1. FASE DE INVESTIGACIÓN

Ninguna de las ONG'S cuenta con un departamento de sistemas actualmente, por ende, se requiere apoyo en la optimización de costos, para ello tener una implementación en la nube, que reduce los costos de inversión inicial del proyecto, y permiten tener el sitio altamente disponible, y no dependa de un dispositivo físico, para funcionar.

1.1. Antecedentes de la institución

- Asociación Jepa Guatemala, Nace de la visión de Alberto Ciferri, químico y biólogo italoamericano, con el deseo de generar desarrollo en las comunidades aledañas a la Antigua Guatemala, con el apoyo de la Fundación Jeppa Limmat, que él lidera actualmente, con sede en Suiza y participación en otros países del mundo.
- Asociación Comunitaria de Proyectos de Desarrollo- Asocoprode -ONG, es una organización, sin ánimos de lucro, ni fines religiosos, trabaja proyectos de desarrollo para fortalecer el sector social y promover el desarrollo de los derechos sociales y civiles de las comunidades más vulnerables y desprotegidas de nuestro país.

1.1.1. Reseña histórica

Asociación Jepa Guatemala, liderada por Alberto Ciferri, comenzó su andadura cuando él llegó a Guatemala en 2003. Desde entonces, la organización ha estado activa en promover la seguridad alimentaria y nutricional,

AGRADECIMIENTOS A:

**Universidad de San
Carlos de Guatemala**

Mi respeto y gratitud por permitirme desarrollar mis habilidades y adquirir conocimientos que serán la base de mi desarrollo profesional, sobre todo por fomentar la conciencia social.

Mis asesores

Por la confianza y el apoyo durante todo mi EPS, al ofrecerme su tiempo y conocimiento para concluir esta etapa.

Mis amigos

Por su apoyo y aprecio brindado, por las experiencias compartidas y su amistad desinteresada.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES.....	III
LISTA DE SÍMBOLOS.....	V
GLOSARIO	VII
RESUMEN	IX
OBJETIVOS.....	XI
INTRODUCCIÓN	XIII
1. FASE DE INVESTIGACIÓN	1
1.1. Antecedentes de la institución.....	1
1.1.1. Reseña histórica	1
1.1.2. Misión	2
1.1.3. Visión.....	2
1.1.4. Servicios que realiza.....	3
1.2. Descripción de las necesidades.....	3
1.2.1. Módulo Landing Page.....	3
1.2.2. Módulo de Inicio de Sesión.....	4
1.2.3. Módulo de Finanzas.....	4
1.2.4. Módulo de calendario y eventos.....	5
1.2.5. Módulo de programa.....	5
1.2.6. Módulo de proyectos.....	5
1.2.7. Módulo de Usuarios.....	5
1.2.8. Módulo de miembros y colaboradores	6
1.2.9. Módulo de socios o donadores	6
1.2.10. Publicación en la Nube.....	6
1.2.11. Gestión de dominio	6
1.3. Priorización de las necesidades.....	7

2.	FASE TÉCNICO PROFESIONAL.....	9
2.1.	Descripción del proyecto.....	10
2.2.	Investigación preliminar para la solución del proyecto.....	11
2.2.1.	Análisis FODA.....	11
2.2.2.	Metodología de trabajo	12
2.2.3.	Plan de contingencia.....	13
2.3.	Presentación de la solución al proyecto	14
2.3.1.	Tecnologías.....	14
2.3.2.	Arquitectura del sistema.....	16
2.4.	Costos del proyecto.....	17
2.4.1.	Recursos humanos.....	17
2.4.2.	Recursos materiales.....	18
2.4.3.	Costos	19
2.5.	Beneficios del proyecto	19
3.	FASE ENSEÑANZA APRENDIZAJE	21
3.1.	Capacitación propuesta.....	21
3.2.	Material elaborado.....	21
	CONCLUSIONES	23
	RECOMENDACIONES	25
	REFE RENCIAS	27
	APÉNDICES	29

ÍNDICE DE ILUSTRACIONES

FIGURAS

Figura 1.	Diagrama de arquitectura.....	17
------------------	-------------------------------	----

TABLAS

Tabla 1.	Riesgos y plan de acción del proyecto.....	14
Tabla 2.	Costos.....	19

LISTA DE SÍMBOLOS

Símbolo	Significado
Q	Quetzales

GLOSARIO

Backend	Aplicación de software encargado de gestionar la información, lógica y manipulación de los datos, es la parte del desarrollo encargada de la lógica para que una aplicación web funcione correctamente.
Sistema	En el contexto de la informática, un sistema se refiere a un conjunto de componentes interconectados que trabajan juntos para realizar un conjunto de funciones.
Despliegue	Aplicación de software encargado de gestionar la información, lógica y manipulación de datos, es la parte del desarrollo encargada de la lógica para que una aplicación web funcione correctamente.
Hardware	El hardware se refiere a los componentes físico de un sistema informático. Esto incluye dispositivos internos como el procesador, la memoria RAM, el disco duro y dispositivos externos como el teclado y entre muchos otros
HTTPS	Es una versión más segura de HTTP, Añade cifrado a través DE SSL para proteger la información que se envía, asegurando que solo el cliente y el servidor puedan entenderla.

HTTP

Es el protocolo básico que permite enviar datos en Internet, facilitando la comunicación entre navegadores web y servidores mediante solicitudes y respuestas.

Página Web

Una página web es un documento digital accesible a través de Internet, diseñada para ser visualizada en un navegador.

ONG

Una organización no gubernamental (ONG) es una entidad que tiene como finalidad trabajar en pro de una causa social, humanitaria, ambiental o de desarrollo, sin estar bajo el control ni financiamiento de algún gobierno.

Entidad Relación

Esta herramienta permite representar de manera visual y estructurada las entidades que forman parte de una base de datos, asignándoles un nombre y un conjunto de atributos. Además, se pueden establecer relaciones entre las diferentes entidades, creando así una estructura que refleja cómo están relacionados los datos entre sí.

RESUMEN

Las organizaciones Asociación Comunitaria de Proyectos de Desarrollo (Asocoprode), Jepa Guatemala no cuentan con ningún tipo de plataforma y es por eso por lo que cada una de las actividades que estas realizan las ejecutan de manera manual ya sea por medio de hojas de cálculo o correos electrónicos para poder comunicarse con los distintos donadores o socios que estas poseen.

Esta falta de tecnología ha dificultado en cierta parte la gestión de la información de las organizaciones, así como el seguimiento de los proyectos, programas que estas poseen dando como consecuencia que no logren llegar a todas las personas que estas quisieran ya que al no existir una plataforma no se han logrado añadir en ocasiones nuevos donadores o colaboradores a las organizaciones.

El capítulo 1 contempla la fase de investigación que brinda un contexto de la situación actual tanto de Jepa Guatemala y Asociación Comunitaria de Proyectos de Desarrollo (Asocoprode), así como de todos los módulos a desarrollar en el proyecto explicando a detalle cada uno de ellos.

El capítulo 2 contempla la fase técnico profesional es decir que especifica que herramientas será necesarias para poder ayudar a brindar la solución final del proyecto planteado. Por otra parte, describirá a detalla todos los recursos necesarios para la culminación de este, así como todos los beneficios que las 2 organizaciones obtendrán.

El capítulo 3 contempla la fase de enseñanza y aprendizaje que ayudara así a los usuarios finales, tales como los propios miembros de las organizaciones y sobre todo pues todas aquellas personas que quieran formar parte de cualquiera de las instituciones.

OBJETIVOS

General

Desarrollar una plataforma que sea de utilidad para las ONG'S para que así puedan manejar toda su información de una manera clara y sencilla y así también dar a conocer todos los proyectos que estas desarrollan.

Específicos

1. Permitir a los usuarios ver los resultados en tiempo real de los proyectos, programas y eventos de las ONG'S actuales o pasados de las ONG's.
2. Fomentar a los usuarios ser parte de las ONG's, ya sea para convertirse en colaboradores de esta o también facilitar el proceso cuando estos deseen donar algo para la organización.
3. Facilitar a los miembros de las ONG'S la gestión de su información facilitando así tener un mejor control de quienes quieren formar parte de su organización, quienes quieren hacer un donativo, así como también ayudarlos a llevar un control de sus cuentas.

INTRODUCCIÓN

Actualmente en un mundo tan digitalizado, donde básicamente cada institución o cualquier tipo de empresa cuenta un sistema o plataforma donde muestran a detalle cada uno de sus productos o servicios que estas o estos prestan. A pesar de eso aún existen instituciones que no cuentan con ningún tipo de plataforma o sistema para gestionar su información o simplemente darse a conocer y es por eso que muchas estas veces no logran progresar o simplemente no logran completar ni concretar ninguno de sus objetivos.

Actualmente Jega Guatemala ni Asocoprode cuentan con ningún tipo de plataforma para poder gestionar su información debido a que todo lo trabajan en su mayoría de manera física y el único medio por el que estas se dan a conocer es por medio de redes sociales o a través de los diferentes programas, proyectos, eventos que estas realizan y si alguien o alguna institución o empresa quiere realizar un donativo esta tiene que contactarse directamente con las organizaciones para poder realizar su donativo así como también si alguien quiere agregarse como nuevo miembro o colaborador de las organizaciones debe realizar su trámite directamente con los encargados de las organizaciones.

Es por eso que actualmente se presenta una propuesta para una plataforma auto gestionable donde tanto Asocoprode como Jega podrán así tener su propia plataforma donde tendrán la capacidad de gestionar toda su información para poder así automatizar cada uno de los procesos que estas realizan.

involucrándose significativamente en proyectos como la Gran Cruzada Nacional por la Nutrición. Este enfoque se refleja en iniciativas como el Diplomado de Promotoras en Seguridad Alimentaria en San Juan Alotenango, que busca mejorar la vida de las comunidades mediante educación en nutrición y medio ambiente.

La Asociación Comunitaria de Proyectos de Desarrollo (ASOCOPRODE) fue fundada el 5 de febrero de 2004 con el objetivo de proporcionar ayuda humanitaria y apoyo a personas de escasos recursos en Guatemala. Desde sus inicios, ASOCOPRODE ha estado activamente involucrada en proyectos que abordan necesidades urgentes en comunidades vulnerables, especialmente durante crisis como la pandemia de COVID-19. Durante este periodo, la organización se destacó por donar más de 20,000 libras de verduras a diferentes comunidades del país, ayudando a mejorar la nutrición de estas poblaciones durante tiempos de emergencia. Esta iniciativa refleja el compromiso de ASOCOPRODE con el bienestar y la seguridad alimentaria de los sectores más desfavorecidos.

1.1.2. Misión

Colaborar en construir una sociedad autosostenible, utilizando las capacidades de las personas que habitan en los sectores menos favorecidos para alcanzar el desarrollo de sus comunidades.

1.1.3. Visión

Ser una organización enfocada en el bienestar social, activa y ágil, que responda con proactividad a la capacitación y desarrollo de las comunidades, promoviendo proyectos de vanguardia que ayuden a sus necesidades.

1.1.4. Servicios que realiza

El Centro de Aprendizaje de Lenguas de la Universidad de San Carlos brinda servicios orientados a la enseñanza de idiomas vernáculos y extranjeros, la enseñanza de idiomas también conlleva servicios de certificación de cursos, exámenes de ubicación, ventas de libros, y otras actividades relacionadas a la enseñanza.

1.2. Descripción de las necesidades

La ONG no cuenta con un departamento de sistemas actualmente, por ende, se requiere apoyo en la optimización de costos, para ello tener una implementación en la nube, que reduce los costos de inversión inicial del proyecto, y permiten tener el sitio altamente disponible, y no dependa de un dispositivo físico, para funcionar.

1.2.1. Módulo Landing Page

Se requiere una pantalla de inicio donde se muestre a detalle toda la información de la organización ya sea por medio de fotos y videos junto con una breve descripción de la misma indicando todos los eventos próximos, los programas próximos así como también los todos los programas y eventos anteriores, junto con la posibilidad de poder llenar una solicitud para ser colaborador en la institución dando también dar la opción de que cualquiera pueda presentar la solicitud para poder socio o donador en la organización.

Algo importante en este módulo, es que los visitantes a la plataforma tendrán la capacidad de poder ver todos los colaboradores destacados

pertenecientes a la institución principalmente los que miembros de la Junta Directiva de la Organización.

Por otra parte, dentro de este Módulo de inicio de se podrán visualizar todas las donaciones en especie que sean donadas a la institución.

Otro apartado dentro del Módulo de Inicio es el de poder ver los principales donadores o socios dentro de la organización, en pocas palabras todos los que se mantienen activos.

Por último, pero no menos importante dentro de este módulo se podrán visualizar todos los proyectos pasados, así como también todos los proyectos en curso de la institución.

1.2.2. Módulo de Inicio de Sesión

Se requiere una pantalla dentro del módulo de inicio, que le dé la capacidad a cualquier usuario dentro la plataforma de poder ingresar sesión. Cabe resaltar que dicha plataforma contara únicamente con 2 tipos de usuarios :1) Usuario administrador 2) operador, estos únicamente podrán ingresar a la plataforma ingresando su correo electrónico, así como su respectiva contraseña.

1.2.3. Módulo de Finanzas

Se requiere una pantalla donde únicamente el usuario administrador definido anteriormente tendrá la capacidad de poder administrar cada una de las entradas y salidas de dinero que posee la institución.

1.2.4. Módulo de calendario y eventos

Se requiere una pantalla donde tanto el usuario administrador, como el usuario operador tengan la capacidad de poder administrar todos los eventos de la organización debido a que en este módulo se tendrá la capacidad de poder ver, agregar, editar y eliminar todos los eventos que posee la organización.

1.2.5. Módulo de programa

Se requiere una pantalla donde el usuario Administrador tendrá la capacidad de poder administrar todos los programas que posee la organización, ya que en este módulo se tendrá la capacidad de poder ver, agregar, editar y eliminar todos los programas que posee la organización.

1.2.6. Módulo de proyectos

Se requiere una pantalla donde el usuario Administrador tendrá la capacidad de poder administrar todos los proyectos ya sea pasados o actuales que posee la organización, ya que en este módulo se tendrá la capacidad de poder ver, agregar, editar y eliminar todos los proyectos que ha tenido o va a adquirir la organización.

1.2.7. Módulo de Usuarios

Se requiere una pantalla donde únicamente el usuario administrador tenga la capacidad de poder crear nuevos usuarios en la plataforma, agregando toda la información crucial del mismo, cabe resaltar que este podrá crear únicamente nuevos colaboradores.

1.2.8. Módulo de miembros y colaboradores

Se requiere una pantalla para poder administrar que a los miembros o colaboradores de la organización, en ella se tendrá capacidad de poder editar, eliminar o agregar nuevos miembros o a colaboradores ya pertenecientes a la organización, se tendrá que guardar primordialmente su información a que se dedican, la fecha en la que se unieron a la organización ya que toda esa información será mostrada en el Módulo de Inicio del sistema.

1.2.9. Módulo de socios o donadores

Se requiere una pantalla para poder administrar que a los donadores o socios de la organización, en ella se tendrá capacidad de poder editar, eliminar o agregar nuevos socios o donadores ya pertenecientes a la organización, se tendrá que guardar primordialmente su información a que se dedican, la fecha en la que se unieron a la organización ya que toda esa información será mostrada en el Módulo de Inicio del sistema.

1.2.10. Publicación en la Nube

Se requiere que todos los servicios planteados, así como también todas las vistas sean publicadas en línea. En este caso se espera que esta sea publicada en AWS.

1.2.11. Gestión de dominio

Para poder publicar el sitio web de la organización es necesario gestionar un dominio por lo que se solicita que se realicen las gestiones necesarias para poder adquirirlo y aplicarlo.

1.3. Priorización de las necesidades

La priorización de las actividades a realizar para el desarrollo de la solución se plasma en el listado de actividades del cronograma y el orden de ejecución de ellas, cada una de las actividades contempla su documentación correspondiente, teniendo una planificación con actividades de desarrollo, configuración y documentación en el siguiente orden lógico:

- Configuración de ambiente
- Configuraciones iniciales
- Gestión de dominio
- Publicación en la Nube
- Módulo de Landing Page
- Módulo de Finanzas
- Módulo de donadores o socios
- Módulo de miembros y colaboradores
- Módulo de usuarios
- Módulo de programas
- Módulo de proyectos
- Módulo de Inicio de Sesión

2. FASE TÉCNICO PROFESIONAL

Los artefactos involucrados en la fase técnico profesional abarcan aqueas parte técnicas del proceso de desarrollo, teniendo los siguientes:

- Product Backlog
 - Se tiene un listado de los requerimientos a implementar, teniendo priorizado por el product owner. Este es un insumo que se ha ido depurando durante la fase de investigación.
- Sprint Backlog
 - Con el objetivo de realizar entregas incrementales en periodos de tiempo razonables, se planea lo que se desarrollará en cada sprint para poder aportar valor al usuario.
- Incremento del producto
 - Teniendo el trabajo de un sprint, se genera como resultado el incremento del producto, es decir, una versión integrada con las funcionalidades que se han trabajado durante un periodo de tiempo.

Las reuniones recurrentes para la fase técnica profesional son:

- Revisión del sprint
 - Para presentar al Product Owner los resultados alcanzados y realizando entregas continuas para que vea el avance obtenido del sprint.

- Retrospectiva
 - Con los asesores, para dar visibilidad de lo que se está haciendo bien, y cómo se puede mejorar.

2.1. Descripción del proyecto

Actualmente, ninguna de las dos instituciones cuenta con un departamento dedicado al desarrollo de software. Por ende, se busca realizar un sistema que aborde específicamente cada una de las necesidades operativas y organizacionales de cada institución. Este sistema será diseñado con una arquitectura que no solo busca optimizar recursos tecnológicos, sino también ayudar a potenciar la eficiencia y efectividad de cada una de las instituciones.

Por otra parte, los fundamentos de este proyecto se basan en crear y desarrollar un sistema adaptable que permita a cada institución gestionar actividades y datos relacionados con donantes y beneficiarios de una manera mucho más eficaz, ayudando así a mejorar significativamente la recopilación de información y, por ende, a cada una de las instituciones a tomar decisiones más informadas. Todo esto será posible gracias a la implementación del sistema.

Este sistema será diseñado para ser intuitivo y fácil de usar, para que los usuarios que interactúen con este sistema no enfrenten una curva de aprendizaje muy pronunciada. Es importante mencionar que la implementación en la nube facilitará que cada una de las actualizaciones y el mantenimiento del sistema se realicen sin interrupciones significativas en el servicio.

Por otra parte, la seguridad de los datos es muy importante en un sistema informático, por ende, esta es una prioridad en el desarrollo del sistema, tomando medidas robustas para proteger la información sensible de cada uno de los

usuarios que interactúen con este sistema. Esto está totalmente alineado con la infraestructura de seguridad avanzada proporcionada por los servicios en la nube.

2.2. Investigación preliminar para la solución del proyecto

Se realizan reuniones periódicas con los interesados, principalmente con el Product Owner para la toma de requerimientos, se realiza una revisión de los requerimientos con el Scrum Máster para delimitar el alcance de cada uno de los requerimientos que serán transformadas en historias de usuario que se desarrollan en la fase técnica, es el insumo que tiene como objetivo el definir los requerimientos del negocio.

2.2.1. Análisis FODA

- Análisis Interno
 - Fortalezas
 - Se cuenta con experto, en desarrollo para poder implementar el proyecto
 - El proyecto será implementado con tecnologías relativamente nuevas, por lo que se podrá contar con documentación actualizada acerca de las tecnologías con las que se trabajará el proyecto.
 - El proyecto será implementado en la nube por lo que podrá ser auto gestionable de una manera sencilla desde cualquier parte del mundo.

- Debilidades
 - Actualmente no se cuenta con ningún tipo de infraestructura dentro de las organizaciones, por lo que todo tiene que trabajarse desde cero.
 - Debido a que tampoco se cuenta con un departamento de sistemas, posteriormente no habrá alguien que le de mantenimiento a la página.
- Oportunidades
 - Crecimiento tecnológico.
 - Proceso de Innovaciones.
 - Sistema protegido.

2.2.2. Metodología de trabajo

La metodología a utilizar será un enfoque basado en Scrum, los principales roles involucrados son:

- Product Owner - Lic. Maria Alejandra Muñoz Leiva
 - Es el responsable de expresar los requerimientos y definir qué trabajos se requieren, es el encargado de gestionar las prioridades y es el representante del negocio.
- Scrum Master - Ing. Luis Estuardo González Ortíz
 - Cumple la función de apoyar al equipo y a la organización enfocándose en el correcto desempeño del equipo y teniendo una visión del proyecto para presentar los requerimientos en historias.

- Team Juan Diego Alvarado
 - El equipo de desarrollo cuenta con el conocimiento técnico que satisface los requerimientos y entrega de resultados comprometidos en intervalos de tiempo definidos.

Las reuniones recurrentes para la fase técnica profesional realizadas son:

- Revisión del sprint
 - Para presentar al Product Owner los resultados alcanzados y realizando entregas continuas para que vea el avance obtenido del sprint.
- Retrospectiva
 - Con los asesores, para dar visibilidad de lo que se está haciendo bien, y cómo se puede mejorar.

2.2.3. Plan de contingencia

Se implementarán medidas para garantizar la continuidad del proyecto, incluyendo documentación técnica y capacitaciones para el mantenimiento.

Tabla 1.

Riesgos y plan de acción del proyecto

Riesgos	Nivel	Plan de acción
Personal asignado al mantenimiento del proyecto.	Alto	Se elaborará un informe detallado sobre cómo funciona el sistema para que cualquier persona encargada del mantenimiento del proyecto pueda utilizarlo de manera más efectiva.
Capacidad de procesamiento y almacenamiento de servidores	Medio	Se busca implementar una solución óptima para poder implementar todos los módulos en la nube.
Difundir el sistema y capacitaciones al personal sobre el uso del sistema	Medio	Se crearán documentos informativos con el objetivo de explicar de manera más clara y detallada cómo funciona el sistema.

Nota. Detalle de plan de contingencia del proyecto. Elaboración propia, realizado con Excel.

2.3. Presentación de la solución al proyecto

Se desarrolló una solución basada en una arquitectura serverless utilizando AWS, optimizando escalabilidad y costos. React y Node.js permiten una experiencia fluida y eficiente, mientras que DynamoDB y S3 garantizan almacenamiento seguro y de alto rendimiento. La integración con CloudFront y API Gateway mejora la entrega de contenido y la gestión de solicitudes.

2.3.1. Tecnologías

Dentro de las tecnologías a utilizar en el proyecto se encuentran:

- Frontend- React
 - React es una biblioteca de JavaScript diseñada para crear interfaces de usuario de manera declarativa y basada en componentes, lo que simplifica el desarrollo de aplicaciones de página única.

- Backend - NodeJS
 - NodeJS es un entorno de tiempo de ejecución para JavaScript, utiliza un modelo de entrada y salida sin bloqueo controlado por eventos, permite ser ligero, eficiente, escalable.

- S3 AWS
 - Amazon S3 (Simple Storage Service) es un servicio de almacenamiento de objetos altamente escalable y seguro, diseñado para almacenar y recuperar grandes volúmenes de datos de manera confiable.

- Amazon CloudFront
 - Amazon CloudFront es una red de entrega de contenido (CDN) que se integra perfectamente con Amazon S3, ofreciendo una manera eficiente de distribuir contenido a los usuarios finales con baja latencia y altas velocidades de transferencia. Mientras que S3 AWS proporciona un servicio de almacenamiento de objetos seguro, duradero y escalable para cualquier cantidad de datos, CloudFront acelera la entrega de estos datos almacenados en S3 a los usuarios finales.

- AWS Lambda y Amazon API Gateway
 - AWS Lambda y Amazon API Gateway combinan sus capacidades para proporcionar una solución potente y escalable para el manejo de solicitudes y la entrega de contenido. AWS Lambda permite ejecutar código en respuesta a eventos, sin la necesidad de administrar servidores, lo que optimiza los recursos y reduce los

costos operativos. Amazon API Gateway actúa como la puerta de entrada para las APIs, gestionando y procesando las solicitudes HTTP antes de dirigirlas a las funciones Lambda correspondientes. Esta integración facilita el desarrollo de aplicaciones serverless que son altamente eficientes y escalables, permitiendo a los desarrolladores centrarse más en la lógica de la aplicación y menos en la infraestructura subyacente.

- **AWS Route 53**
 - Amazon Route 53 es un servicio de sistema de nombres de dominio (DNS) escalable y altamente disponible que facilita la conexión entre los usuarios y las aplicaciones alojadas en AWS. Route 53 gestiona el enrutamiento del tráfico de Internet al traducir nombres de dominio en direcciones IP, garantizando un rendimiento rápido y confiable.
- **AWS Dynamo DB**
 - Amazon DynamoDB es una base de datos NoSQL completamente gestionada que ofrece rendimiento de baja latencia y escalabilidad automática para manejar grandes volúmenes de datos y solicitudes

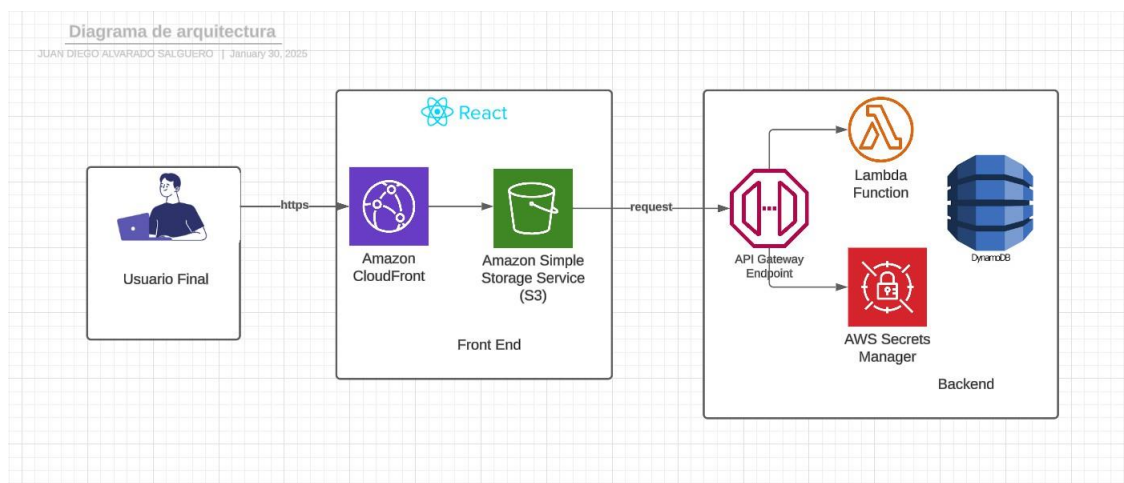
2.3.2. Arquitectura del sistema

La arquitectura implementada busca solventar los requerimientos de implementación de la biblioteca virtual. Esta arquitectura de software se seleccionó por su capacidad para modularizar el sistema, facilitando el desarrollo y despliegue de nuevas características de manera independiente, garantizar la escalabilidad y robustez del proyecto, su enfoque basado en microservicios y el

uso de contenedores proporciona una base sólida y flexible para el crecimiento futuro del sistema.

Figura 1.

Diagrama de arquitectura



Nota. Diagrama de arquitectura del sistema de biblioteca virtual. Elaboración propia, realizado con Drawio.

2.4. Costos del proyecto

Es importante tomar en cuenta los recursos involucrados en el desarrollo del sistema, tanto recursos humanos como recursos materiales para poder plantear un desglose de costos del proyecto.

2.4.1. Recursos humanos

- Juan Diego Alvarado: Estudiante con cierre en la carrera de Ingeniería Ciencias y Sistemas, encargado del desarrollo del proyecto de EPS y su respectiva documentación, técnica y de usuario.
- Ing. Luis Fernando Cajas Calijau: Asesor del proyecto de EPS por parte de la Escuela de Ciencias y Sistemas de la Universidad de San Carlos de Guatemala.
- Lic. Jordy Noriega: Encargado de la supervisión del proyecto por parte de la ONG Jepa Guatemala
- Josué Misael Chacon Barahona: Encargado de la supervisión del proyecto por parte de la ONG Asocoprode

2.4.2. Recursos materiales

- Computadora portátil
- Micrófono
- Conexión a Internet de alta velocidad
- Servicio de luz eléctrica
- Lugar de trabajo adecuado
- Cámara web de alta definición para videoconferencias
- Auriculares con cancelación de ruido para mejorar la calidad del audio en reuniones
- Plataformas de almacenamiento en la nube para compartir archivos.
- Dispositivos móviles (smartphones o tabletas) para acceso y gestión remota
- Smartphone
- Teclado

2.4.3. Costos

El proyecto requiere diversos recursos para su ejecución, incluyendo tiempo de implementación y asesoría técnica, así como equipamiento esencial para el desarrollo y pruebas del proyecto.

Tabla 2.

Costos

Recursos	Cantidad	Costo Unitario	Subtotal
Tiempo de implementación	de 600 hrs	Q. 100.00	Q 60,000.00
Tiempo de Asesoría de parte del Ing. Luis Fernando Cajas	50 hrs	Q 150.00	Q 7,500.00
Lugar de trabajo	6 meses	Q 1,000.00 mensual	Q 1000.00
Servicio de internet	6 meses	Q 500.00 mensual	Q 3,000.00
Servicio de luz	6 meses	Q 300.00 mensual	Q 1,800.00
TOTAL			Q 90,800.00

Nota. Detalle del presupuesto para la realización del proyecto de investigación. Elaboración propia.

2.5. Beneficios del proyecto

- Uno de los principales beneficios que obtendrán las instituciones es que su información estará resguardada de manera segura.
- Además, un beneficio importante es la capacidad de darse a conocer de forma más sencilla, lo cual les permitirá tener un mejor control sobre quiénes desean formar parte de estas instituciones.

- Gracias a módulos especializados, cada una de las instituciones pueden gestionar finanzas, usuarios, programas y proyectos de manera centralizada, reduciendo la carga operativa.
- La plataforma está diseñada para crecer junto con la organización, adaptándose a sus necesidades y garantizando un alto rendimiento y estabilidad a medida que aumenta el número de usuarios. Su infraestructura escalable permite manejar un mayor volumen de datos y transacciones sin comprometer la velocidad ni la experiencia del usuario. Además, está optimizada para ajustarse a distintos niveles de demanda, asegurando que los recursos en la nube se utilicen de manera eficiente y evitando costos innecesarios. De esta forma, las instituciones pueden expandir sus operaciones con confianza, sabiendo que la plataforma seguirá funcionando de manera óptima sin importar su crecimiento.

3. FASE ENSEÑANZA APRENDIZAJE

Para llevar a cabo la fase de enseñanza de manera efectiva, se han planificado varias acciones. Una de ellas es la creación de tutoriales que muestren de manera sencilla y clara el correcto funcionamiento del proyecto. Además, se busca elaborar una documentación detallada para explicar el código y cada una de las funciones principales del proyecto.

3.1. Capacitación propuesta

- Reuniones presenciales o virtuales: Organización de reuniones, ya sean presenciales o virtuales, según la conveniencia de cada institución, para transferir el conocimiento a todas las partes involucradas en el proyecto.
- Manuales técnicos: Elaboración de manuales técnicos que expliquen el correcto despliegue del proyecto. Además, estos manuales detallarán cada una de las funcionalidades principales del proyecto, proporcionando un soporte adecuado al código.
- Documentación de servicios: Documentación exhaustiva de cada uno de los servicios implementados en el proyecto.

3.2. Material elaborado

De la mano de la capacitación propuesta, se trabajan diferentes materiales de capacitación y soporte.

- Videotutorial sistema funcional explicando cada módulo del sistema.
- Presentación de cada uno de los módulos del sistema.

- Videotutorial del despliegue del proyecto.
- Documento con especificaciones técnicas del sistema, documentación tradicional.

CONCLUSIONES

1. Permitir a los usuarios ver en tiempo real el avance sobre los proyectos, programas y eventos de las organizaciones actuales o pasados es crucial para el funcionamiento y éxito de las organizaciones. Al mantener a los usuarios o visitantes de la plataforma informados sobre todo lo que realizan las organizaciones ayuda a crear una mayor conciencia sobre los problemas y las causas que estas organizaciones apoyan. Esto puede generar más apoyo y participación por parte de las personas que se sientan atraídas hacia las organizaciones, ya que las personas al ver esa información pueden conocer mejor la importancia y el impacto del trabajo que las organizaciones.
2. Facilitarse el proceso a las personas de poder ser nuevos miembros de las organizaciones, puede dar como resultado que muchas más personas se sientan interesadas en ser parte de las organizaciones debido a que tanto un colaborador, como un donador son de suma importancia para que estas organizaciones puedan tener éxito y continúen existiendo en el tiempo.
3. La información es un recurso clave para que el trabajo de cualquier organización funcione correctamente y logre cumplir sus objetivos a corto o a mediano plazo. Tener una buena gestión de la información puede facilitar en gran manera tener un mejor control de quienes quieren ser parte de las instituciones.

RECOMENDACIONES

1. Manejar un control de versiones, dentro del proyecto es de suma importancia ya que este tiene la capacidad de realizar pruebas y experimentos sin afectar la versión principal del código fuente. Si se comete un error, es posible retroceder a una versión anterior y corregirlo sin afectar el trabajo de los demás colaboradores.
2. Es crucial garantizar la seguridad de los datos, especialmente si estos contienen información sensible de los usuarios. Por lo tanto, se debe verificar que la plataforma seleccionada cuente con medidas de seguridad sólidas y adecuadas para proteger los datos de los usuarios de cualquier posible amenaza o riesgo.
3. Es fundamental brindar capacitación y soporte a los usuarios para que puedan utilizar la plataforma de manera efectiva y solucionar cualquier problema que puedan encontrar. Esto permitirá que los usuarios se sientan cómodos y confiados al utilizar la plataforma, lo que a su vez aumentará la: satisfacción del usuario y mejorará la eficiencia en la realización de sus tareas en la plataforma. Además, también se deben proporcionar recursos útiles y una asistencia adecuada para cualquier problema o inquietud que puedan surgir en el futuro.

4. Es importante considerar la necesidad de un sistema escalable, ya que a medida que aumenta el volumen de usuarios en la plataforma, los costos en la nube pueden incrementarse. Por ello, es fundamental evaluar y optimizar el uso de los recursos para garantizar un equilibrio entre rendimiento y costos operativos.

REFE RENCIAS

coi

Amazon Web Services, Inc. (s.f.). *¿Qué es una aplicación web?* AWS.
<https://aws.amazon.com/es/what-is/web-application/>

Amazon Web Services, Inc. (s.f.). *¿Qué es una interfaz de programación de aplicaciones (API)?* AWS. <https://aws.amazon.com/es/what-is/api/>

Amazon Web Services, Inc. (s.f.). *Amazon API Gateway: Fully managed API gateway service.* <https://aws.amazon.com/api-gateway/>

Amazon Web Services, Inc. (s.f.). *Amazon CloudFront: Content delivery network (CDN) service.* <https://aws.amazon.com/cloudfront/>

Amazon Web Services, Inc. (s.f.). *Amazon Route 53: Scalable domain name system (DNS) web service.* <https://aws.amazon.com/route53/>

Amazon Web Services, Inc. (s.f.). *Amazon S3: Scalable storage in the cloud.* <https://aws.amazon.com/s3/>

Amazon Web Services, Inc. (s.f.). *AWS Lambda: Run code without thinking about servers or clusters.* <https://aws.amazon.com/lambda/>

Facebook, Inc. (s.f.). *React: Declarative, efficient, and flexible JavaScript library for building user interfaces.* <https://react.dev/>

OpenJS Foundation. (s.f.). *Node.js: JavaScript runtime built on Chrome's V8 JavaScript engine*. <https://nodejs.org/>

APÉNDICES

Apéndice 1.

Manual técnico – Backend

Manual Técnico - Backend

Introducción

Este documento detalla la configuración e implementación del backend del proyecto **jepaBack**, utilizado en **Asocoprode** y **Jepa Guatemala**. Se emplea **Serverless Framework v4** para desplegar funciones en AWS Lambda, junto con API Gateway y DynamoDB.

Resumen

El backend de **jepaBack** utiliza **AWS Lambda** y **DynamoDB** para gestionar operaciones sin servidor. Se definen funciones CRUD para entidades como **colaboradores, donadores, operadores, administradores, programas y proyectos**. La configuración de la infraestructura se maneja mediante el archivo `serverless.yml`, el cual define los servicios, los recursos de base de datos y las políticas de seguridad.

Continuación Apéndice 1.

 **Configuración Inicial**

El backend se configura a través de **Serverless Framework**, utilizando un archivo `serverless.yml` que define los servicios, recursos y permisos necesarios.

◆ **Requisitos Previos**

Antes de desplegar el backend, asegúrate de tener instalado:

-  **Node.js 18.x**
-  **AWS CLI** configurado con las credenciales correctas
-  **Serverless Framework v4** instalado globalmente:

```
npm install -g serverless
```

Continuación Apéndice 1.

Estructura del Proyecto

El backend está organizado en carpetas de funciones específicas para cada entidad. La estructura del código es la siguiente:

```
backend/  
├─ build/  
│   └─ functions/  
│       └─ authenticate/  
│           ├── getUserInfo.ts  
│           └─ login.ts  
│       └─ operators/  
│           ├── createOperator.ts  
│           ├── getAllOperators.ts  
│           ├── updateOperator.ts  
│           └─ deleteOperator.ts  
│       └─ admins/  
│           ├── createAdmin.ts  
│           ├── getAllAdmins.ts  
│           ├── updateAdmin.ts  
│           └─ deleteAdmin.ts  
│       └─ donors/  
│           ├── createDonor.ts  
│           ├── getAllDonors.ts  
│           ├── updateDonor.ts  
│           └─ deleteDonor.ts  
├─ serverless.yml  
├─ package.json  
└─ .env
```

Continuación Apéndice 1.

Configuración del `serverless.yml`

El archivo `serverless.yml` contiene la configuración del backend, incluyendo la definición de recursos de **DynamoDB**, API Gateway y permisos.

Resumen del `serverless.yml`

El archivo `serverless.yml` configura el backend **jepaBack**, definiendo los servicios, recursos y permisos en **AWS** con **Serverless Framework v4**.

✦ Configuraciones Generales

- **Runtime:** `nodejs18.x`
- **Región:** `us-east-1`
- **Permisos:** IAM roles para acceso a DynamoDB y S3
- **Variables de entorno:** Cargadas desde `.env`
- **Plugins utilizados:** `serverless-offline`, `serverless-plugin-typescript`

✦ Recursos Definidos

- **DynamoDB Tables** para almacenar datos de:
 - `CollaboratorsTable`
 - `DonorsTable`
 - `OperatorsTable`
 - `AdminsTable`
 - `ProgramsTable`
 - `ProjectsTable`
- **Índices secundarios globales (GSI)** en DynamoDB para consultas optimizadas (Ej: `EmailIndex`).
- **API Gateway** con rutas REST para cada entidad.
- **CORS habilitado** para permitir solicitudes desde frontend.

Continuación Apéndice 1.

🔥 Funciones CRUD en AWS Lambda

Cada entidad tiene funciones específicas con validación mediante **Zod** y autenticación **JWT**.

Entidad	Funciones Implementadas
Colaboradores	<code>createCollaborator</code> , <code>getAllCollaborators</code> , <code>updateCollaborator</code> , <code>deleteCollaborator</code>
Donadores	<code>createDonor</code> , <code>getAllDonors</code> , <code>updateDonor</code> , <code>deleteDonor</code>
Operadores	<code>createOperator</code> , <code>getAllOperators</code> , <code>updateOperator</code> , <code>deleteOperator</code>
Administradores	<code>createAdmin</code> , <code>getAllAdmins</code> , <code>updateAdmin</code> , <code>deleteAdmin</code>
Programas	<code>createProgram</code> , <code>getAllPrograms</code> , <code>updateProgram</code> , <code>deleteProgram</code>
Proyectos	<code>createProject</code> , <code>getAllProjects</code> , <code>updateProject</code> , <code>deleteProject</code>

🔥 Despliegue y Comandos Útiles

- Desplegar backend en AWS:

```
serverless deploy
```

Continuación Apéndice 1.

Base de Datos DynamoDB

DynamoDB es una base de datos NoSQL altamente escalable y sin servidor utilizada en **jepaBack**. Cada entidad principal tiene su propia tabla en **DynamoDB**, y algunas incluyen índices secundarios globales para facilitar consultas eficientes.

Ejemplo de definición de una tabla en `serverless.yml`:

```
resources:
  Resources:
    CollaboratorsTable:
      Type: AWS::DynamoDB::Table
      Properties:
        TableName: CollaboratorsTable
        AttributeDefinitions:
          - AttributeName: id
            AttributeType: S
          - AttributeName: email
            AttributeType: S
        KeySchema:
          - AttributeName: id
            KeyType: HASH
        GlobalSecondaryIndexes:
          - IndexName: EmailIndex
            KeySchema:
              - AttributeName: email
                KeyType: HASH
            Projection:
              ProjectionType: ALL
        BillingMode: PAY_PER_REQUEST
```

Cada tabla tiene:

- **id** como clave primaria
- **Atributos adicionales** según la entidad
- **Índices secundarios globales** para búsquedas optimizadas
- **Modo de facturación PAY_PER_REQUEST** para una mejor escalabilidad

Continuación Apéndice 1.

CRUD en Lambdas

El backend implementa operaciones CRUD para múltiples entidades, asegurando validaciones con **Zod** y seguridad con **JWT**.

```
### 1. Creación de un colaborador

/**
 * ts
 */
export const handler: APIGatewayProxyHandler = async (event) => {
  const headers = applyCorsHeaders(event);
  try {
    const token = event.headers['Authorization']?.split(' ')[1];
    if (!token || !(await validateToken(token, false))) {
      return { statusCode: 403, headers, body: JSON.stringify({ error: 'Invalid token' }) };
    }

    const data = JSON.parse(event.body || '{}');
    const validatedData = collaboratorSchema.parse(data);

    const id = uuidv4();
    const createdAt = getGuatemalaTime();

    await dynamoDb.put({ TableName: 'CollaboratorsTable', Item: { id, ...validatedData, createdAt } }).promise();
    return { statusCode: 201, headers, body: JSON.stringify({ message: 'Collaborator created', id }) };
  } catch (error) {
    return { statusCode: 400, headers, body: JSON.stringify({ error: error.message }) };
  }
};
```

Explicación:

- **Autenticación:** Extrae el token desde los headers y lo valida con `validateToken(token, false)`.
- **Validación de datos:** Parsea el cuerpo del request usando `collaboratorSchema.parse(data)`.
- **Generación de ID:** Usa `uuidv4()` para generar un `id` único.
- **Registro en DynamoDB:** Inserta el nuevo colaborador en la tabla `CollaboratorsTable`.
- **Respuesta:** Retorna un código `201` con un mensaje de éxito o `400` si hay errores en la validación.

Continuación Apéndice 1.

2. Actualización de una donación

```
export const handler: APIGatewayProxyHandler = async (event): Promise<APIGatewayProxyResult> => {
  const headers = applyCorsHeaders(event);
  const { id } = event.queryStringParameters || {};
  if (!id) {
    return { statusCode: 400, headers, body: JSON.stringify({ error: 'ID is required to update donation' }) }
  }

  try {
    const data = JSON.parse(event.body || '{}');
    const validatedData = donationUpdateSchema.parse(data);
    const updateExpression = [];
    const expressionAttributeNames: Record<string, string> = {};
    const expressionAttributeValues: Record<string, any> = {};

    for (const [key, value] of Object.entries(validatedData)) {
      updateExpression.push(`#${key} = :${key}`);
      expressionAttributeNames[`#${key}`] = key;
      expressionAttributeValues[`:${key}`] = value;
    }

    const params = {
      TableName: 'DonationsTable',
      Key: { id },
      UpdateExpression: `SET ${updateExpression.join(', ')}`,
      ExpressionAttributeNames: expressionAttributeNames,
      ExpressionAttributeValues: expressionAttributeValues,
      ReturnValues: 'ALL_NEW',
    };

    const result = await dynamoDb.update(params).promise();
    return { statusCode: 200, headers, body: JSON.stringify({ message: 'Donation updated successfully', data: result }) }
  } catch (error) {
    return { statusCode: 400, headers, body: JSON.stringify({ error: 'Could not update donation: ${error}' }) }
  }
};
```

Explicación:

- **Autenticación:** Valida el token de autorización (`Authorization: Bearer <token>`).
- **Verificación del id de la donación:** Extrae el `id` desde `queryStringParameters` y verifica que sea válido.
- **Validación de datos:** Usa `donationUpdateSchema.parse(data)` para asegurar que los datos sean correctos.
- **Construcción dinámica de la actualización:**
 - Usa `UpdateExpression` para actualizar solo los campos provistos en la solicitud.
 - Define `ExpressionAttributeNames` y `ExpressionAttributeValues` para evitar conflictos con palabras reservadas.
- **Actualización en DynamoDB:** Modifica los datos en `DonationsTable` usando `update()`.
- **Respuesta:**
 - Devuelve un código `200` si la actualización es exitosa.
 - Devuelve `400` si hay errores en la validación de datos.
 - Devuelve `500` si ocurre un error inesperado en la base de datos.

Continuación Apéndice 1.

3. Obtención de todos los colaboradores

```
export const handler: APIGatewayProxyHandler = async (event): Promise<APIGatewayProxyResult> => {
  const corsHeaders = applyCorsHeaders(event);
  try {
    const params = { TableName: 'CollaboratorsTable' };
    const result = await dynamoDb.scan(params).promise();
    return { statusCode: 200, headers: corsHeaders, body: JSON.stringify(result.Items) };
  } catch (error) {
    return { statusCode: 500, headers: corsHeaders, body: JSON.stringify({ error: 'Could not retrieve coll'
  }
};
```

Explicación:

- **Autenticación:** Valida el token con `validateToken(token, false)`.
- **Consulta en DynamoDB:** Realiza un `scan()` en la tabla `CollaboratorsTable` para obtener todos los registros.
- **Respuesta:**
 - Devuelve un código `200` con la lista de colaboradores.
 - Devuelve `500` si hay un error en la consulta.

4. Eliminación de un donador

```
export const handler: APIGatewayProxyHandler = async (event): Promise<APIGatewayProxyResult> => {
  const headers = applyCorsHeaders(event);
  const { id } = event.queryStringParameters || {};
  if (!id) {
    return { statusCode: 400, headers, body: JSON.stringify({ error: 'ID is required to delete donor' }) }
  }

  try {
    await dynamoDb.delete({ TableName: 'DonorsTable', Key: { id } }).promise();
    return { statusCode: 200, headers, body: JSON.stringify({ message: 'Donor with ID ${id} deleted succes'
  } catch (error) {
    return { statusCode: 500, headers, body: JSON.stringify({ error: 'Could not delete donor with ID ${id}
  }
};
```

Explicación:

- **Autenticación:** Valida el token de autorización.
- **Verificación del id:** Extrae el `id` de `queryStringParameters` y verifica que sea válido.
- **Eliminación en DynamoDB:**
 - Usa `delete()` para borrar el donador en `DonorsTable` basado en su `id`.
- **Respuesta:**
 - Devuelve un código `200` si la eliminación es exitosa.
 - Devuelve `400` si el `id` no es proporcionado.
 - Devuelve `500` si ocurre un error en la base de datos.

Continuación Apéndice 1.

Conclusión

Estas funciones siguen el mismo flujo básico:

1. **Validación de token JWT** → Se verifica la autenticidad del usuario.
2. **Validación de datos con Zod** → Se asegura que los datos enviados sean correctos.
3. **Interacción con DynamoDB** → Se realizan las operaciones necesarias (inserción, actualización, consulta o eliminación).
4. **Manejo de respuestas HTTP** → Se retornan códigos `200`, `201`, `400`, `401`, `403`, o `500` dependiendo del resultado.

Despliegue y Comandos Útiles

Para desplegar el backend en **AWS**, ejecuta:

```
serverless deploy
```

Para correr el backend de manera local:

```
serverless offline
```

El `package.json` contiene comandos clave para la ejecución del backend:

```
"scripts": {  
  "build": "tsc",  
  "build:watch": "concurrently \"npm run build -- --watch\" \"serverless offline\"",  
  "offline": "concurrently \"npm run build\" \"serverless offline\"",  
  "deploy": "npm run build && serverless deploy"  
}
```

✦ ****Resumen de Funciones Clave****

Este resumen detalla las funciones principales utilizadas en el backend, incluyendo validación de archivos, autenticación JWT, manejo de fechas y seguridad CORS.

Continuación Apéndice 1.

🔑 **Resumen de Funciones Clave**

Este resumen detalla las funciones principales utilizadas en el backend, incluyendo validación de archivos, autenticación JWT, manejo de fechas y seguridad CORS.

🔍 1. Validación de Tipo de Archivo (isValidFileType)

```
export function isValidFileType(file: Blob): boolean {
  const validMimeTypes = [
    'application/pdf',
    'image/jpeg',
    'image/png',
    'image/gif',
    'image/bmp',
    'image/webp',
  ];
  return validMimeTypes.includes(file.type);
}### • **Descripción**
```

- Permite verificar si el tipo de archivo es válido antes de procesarlo.
- Se aceptan archivos PDF e imágenes en formatos **JPEG, PNG, GIF, BMP y WEBP**.
- Devuelve **true** si el archivo es válido, **false** en caso contrario.

🔑 2. Decodificación y Validación de Tokens JWT

```
import * as jwt from 'jsonwebtoken';

const JWT_SECRET = process.env.JWT_SECRET || 'supersecretkey123';

/**
 * Verifies and decodes the JWT token.
 */
const decodeToken = (token: string): { id: string; role: string } | null => {
  try {
    return jwt.verify(token, JWT_SECRET) as { id: string; role: string };
  } catch (error) {
    throw new Error('Invalid or expired token');
  }
};
```

• Descripción

- **decodeToken:**
 - Verifica y decodifica un token JWT.
 - Extrae el **id** y **rol** del usuario.
 - Lanza un error si el token es inválido o ha expirado.

Continuación Apéndice 1.

3. Validación del Usuario en DynamoDB

```
import { DynamoDB } from 'aws-sdk';
const dynamoDb = new DynamoDB.DocumentClient();

/**
 * Checks if the token's user ID exists in either OperatorsTable or AdminsTable.
 */
const validateTokenInDB = async (id: string): Promise<boolean> => {
  const operatorParams = { TableName: 'OperatorsTable', Key: { id } };
  const adminParams = { TableName: 'AdminsTable', Key: { id } };

  try {
    const operatorResult = await dynamoDb.get(operatorParams).promise();
    if (operatorResult.Item) return true;
  } catch (error) {
    console.error('Error fetching operator information:', error);
  }

  try {
    const adminResult = await dynamoDb.get(adminParams).promise();
    if (adminResult.Item) return true;
  } catch (error) {
    console.error('Error fetching admin information:', error);
  }

  return false;
};
```

• Descripción

- Busca el usuario en **OperatorsTable** o **AdminsTable**.
- Si el usuario existe en cualquiera de las tablas, devuelve **true**.
- Maneja errores de consulta y registra fallos en la consola.

4. Validación de Token con Fallback

```
/**
 * Validates the token, checking both database and direct JWT_SECRET validation.
 */
const validateToken = async (token: string, validateDB: boolean = true): Promise<boolean> => {
  if (!token) throw new Error('Authorization token is required');

  const decoded = validateDB ? decodeToken(token) : true;
  if (decoded === true) return true;
  if (!decoded) return false;

  const { id } = decoded;

  if (validateDB) return await validateTokenInDB(id);

  return (await validateTokenInDB(id)) || (token === JWT_SECRET);
};
```

• Descripción

- Verifica el token de **dos maneras**:
 1. **Base de datos**: Comprueba si el **id** existe en DynamoDB.
 2. **Validación directa**: Si falla la verificación en BD, compara el token con **JWT_SECRET**.
- Retorna **true** si el token es válido, **false** si es inválido o no se encuentra en la BD.

Continuación Apéndice 1.

🕒 5. Manejo de Fechas para Guatemala

```
/**
 * Returns the current time in Guatemala timezone (UTC-6).
 */
export const getGuatemalaTime = (): string => {
  const date = new Date();
  const guatemalaOffset = -6 * 60;
  const guatemalaTime = new Date(date.getTime() + guatemalaOffset * 60000);
  return guatemalaTime.toISOString();
};

/**
 * Formats a given UTC date to Guatemala timezone (UTC-6).
 */
export const formatToGuatemalaTime = (utcDate: string): string => {
  const date = new Date(utcDate);
  const guatemalaOffset = -6;
  const guatemalaTime = new Date(date.getTime() + guatemalaOffset * 60 * 60 * 1000);
  return guatemalaTime.toISOString();
};
```

• Descripción

- `getGuatemalaTime`:
 - Obtiene la hora actual de Guatemala (UTC-6).
 - Devuelve la fecha en formato ISO.
- `formatToGuatemalaTime`:
 - Convierte una fecha UTC a la zona horaria de Guatemala.

Continuación Apéndice 1.

6. Middleware de CORS

```
const allowedOrigins = [
  'https://d1outqhl2uvb.cloudfront.net',
  'https://jepaguatemala.com',
  'http://localhost:5173',
];

/**
 * Applies CORS headers to API responses.
 */
export const applyCorsHeaders = (event: any): Record<string, string> => {
  const origin = event.headers?.origin || event.headers?.Origin;
  const isAllowedOrigin = origin && allowedOrigins.includes(origin);

  return {
    'Access-Control-Allow-Origin': isAllowedOrigin ? origin : 'null',
    'Access-Control-Allow-Credentials': 'true',
    'Access-Control-Allow-Headers': 'Content-Type, Authorization, X-API-Key',
    'Access-Control-Allow-Methods': 'OPTIONS, POST, GET, PUT, DELETE',
  };
};
```

• Descripción

- Define los **dominios permitidos** para acceder al backend.
- **Habilita CORS** en las respuestas de API Gateway.
- Permite métodos HTTP como **POST, GET, PUT y DELETE**.

7. Configuración de AWS y S3

```
export default {
  AWS: {
    accessKeyId: process.env.AWS_ACCESS_KEY_ID || '',
    secretAccessKey: process.env.AWS_SECRET_ACCESS_KEY || '',
    region: process.env.AWS_REGION || 'us-east-1'
  },
  S3: {
    uploadBucket: process.env.S3_UPLOAD_BUCKET || 'images-jepa-guatemala'
  }
};
```

• Descripción

- Configura credenciales de **AWS** para acceso a servicios.
- Define el **bucket S3** donde se almacenarán imágenes.

Conclusión

Este manual proporciona una visión completa del backend de , detallando su arquitectura, configuración y despliegue. Además, este backend es utilizado tanto para los proyectos de **Asocoprode** y **Jepa Guatemala**.

Continuación Apéndice 1.

Manual Técnico - Frontend (React)

Introducción

Este documento explica cómo está configurado e implementado el **frontend** del proyecto **jepaBack**, utilizado en **Asocoprode** y **Jepa Guatemala**. El proyecto está construido con **React** y utiliza varias bibliotecas esenciales para facilitar la navegación, la visualización y la autenticación de usuarios. Entre estas bibliotecas se destacan **React Router**, **MUI Joy** y **Context API**.

Aunque la estructura general del proyecto es la misma para **Asocoprode** y **Jepa Guatemala**, existen diferencias notables en la personalización de las imágenes y algunas configuraciones que pueden adaptarse según el entorno de cada proyecto. El proyecto sigue una estructura escalable y modular que facilita la adición de nuevas funcionalidades y el mantenimiento a largo plazo.

El enfoque en **React** permite aprovechar las ventajas de la arquitectura basada en componentes reutilizables, mientras que **React Router** facilita una navegación fluida entre vistas públicas y privadas, asegurando una experiencia de usuario sin interrupciones. **MUI Joy** proporciona componentes visuales consistentes y accesibles, lo que mejora la estética y la usabilidad de la aplicación.

Resumen


El **frontend de jepaBack** está construido con **React** y **TypeScript** para garantizar una mejor experiencia de desarrollo, rendimiento y escalabilidad. El proyecto está diseñado para ser usado por **Asocoprode** y **Jepa Guatemala**, y se adapta a ambos mediante una estructura flexible.

Estructura de Archivos:

```
frontend/
├── src/
│   ├── components/      # Componentes reutilizables
│   ├── pages/           # Páginas públicas y privadas
│   ├── routes/          # Definición de rutas
│   ├── App.tsx          # Componente principal de la aplicación
│   └── AuthProvider.tsx # Proveedor de autenticación
```



La estructura del proyecto permite una fácil navegación y separación de responsabilidades, facilitando la implementación de nuevas características sin comprometer la organización existente.

Continuación Apéndice 1.

 **Configuración Inicial**

El proyecto utiliza **React** y **TypeScript** para garantizar la seguridad de tipos y mejorar la calidad del código. **MUI Joy** se utiliza para la parte visual, mientras que **React Router** gestiona la navegación entre las diferentes vistas del frontend.

Requisitos Previos

-  **Node.js 18.x:** Es necesario para poder ejecutar la aplicación localmente y realizar tareas de construcción.
-  **npm (Node Package Manager):** Se utiliza para instalar las dependencias del proyecto.

Instalación de Dependencias

Para instalar todas las dependencias necesarias para ejecutar el proyecto, debes ejecutar el siguiente comando en tu terminal:

```
npm install
```

Este comando descarga e instala todas las bibliotecas y herramientas necesarias para el proyecto, como **React**, **MUI Joy**, **React Router** y otras dependencias de desarrollo.

Para poder correr el proyecto en desarrollo es necesario correr el siguiente comando:

```
npm run dev
```

Variables de Entorno Necesarias

Para que el proyecto funcione correctamente, es necesario configurar las siguientes variables de entorno en un archivo `.env` en la raíz del proyecto:

```
VITE_BACK_URL=https://udvxa18pu1.execute-api.us-east-1.amazonaws.com/dev
VITE_LINK_S3=https://images-asocoprode.s3.us-east-1.amazonaws.com/
VITE_JWT_SECRET=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJPbmtpbmUgSldUIEJ1Y...
```

Continuación Apéndice 1.



Estructura del Proyecto

El proyecto está dividido en varias carpetas que contienen las diferentes funcionalidades de la aplicación. Esta organización modular facilita la escalabilidad y el mantenimiento de la base de código. A continuación, se describe la estructura de carpetas:

```
frontend/
├── src/
│   ├── assets/           # Archivos estáticos como imágenes
│   ├── AuthProvider/     # Proveedor para la autenticación de usuario
│   ├── components/       # Componentes reutilizables de la interfaz
│   ├── constants/        # Constantes y configuraciones globales
│   ├── functions/        # Funciones utilitarias y helpers
│   ├── images/           # Imágenes específicas del proyecto
│   ├── pages/            # Páginas públicas y privadas
│   │   ├── private/      # Páginas privadas que requieren autenticación
│   │   └── public/       # Páginas públicas accesibles sin autenticación
│   ├── routes/           # Definición de rutas y navegación
│   ├── services/         # Servicios y API de backend
│   ├── theme/            # Personalización del tema visual
│   ├── types/            # Tipos y interfaces de TypeScript
│   ├── .env              # Variables de entorno
│   ├── index.html        # HTML principal
│   ├── package.json      # Configuración del proyecto
│   ├── tsconfig.json     # Configuración de TypeScript
│   └── vite.config.ts    # Configuración de Vite
```

Esta estructura es flexible y permite una fácil escalabilidad, lo que facilita la implementación de nuevas funcionalidades y la organización del proyecto.

Continuación Apéndice 1.

Funciones Generales

1. Enumeraciones (Enums)

Las enumeraciones son utilizadas para definir conjuntos de valores constantes y predefinidos en el sistema. En este caso, las enumeraciones ayudan a estructurar los estados y tipos de donaciones.

1.1 DonationState

```
export enum DonationState {  
  ACTIVE = "active",  
  INACTIVE = "inactive",  
  PENDING = "pending",  
}
```

- Define tres posibles estados para una donación:
 - `ACTIVE`: La donación está activa y disponible en el sistema.
 - `INACTIVE`: La donación está inactiva y no se muestra en listados.
 - `PENDING`: La donación está pendiente de aprobación o procesamiento.

1.2 DonationType

```
export enum DonationType {  
  CASH = "cash",  
  IN_KIND = "in-kind",  
}
```

- Define los dos tipos de donaciones permitidos:
 - `CASH`: Donaciones en efectivo.
 - `IN_KIND`: Donaciones en especie (productos o servicios).
-

Continuación Apéndice 1.

2. Interfaces para Donaciones

Las interfaces definen la estructura de datos utilizada en las donaciones.

```
export interface CreateDonationRequest {  
  name: string;  
  dpi_Passport: string;  
  email: string;  
  phone: string;  
  description: string;  
  photoprrophy?: string;  
  state: "active" | "inactive" | "pending";  
  type: "cash" | "in-kind";  
  visible: boolean;  
}
```

- Esta interfaz define la estructura de los datos necesarios para la creación de una donación.
- `photoprrophy` es opcional, ya que no todas las donaciones requieren una imagen.

```
export interface CreateDonationResponse {  
  message: string;  
  id: string;  
}
```

- Esta interfaz representa la respuesta de la API tras la creación de una donación.
- Incluye un mensaje de confirmación y el identificador único de la donación.

```
export interface UpdateDonationRequest {  
  name?: string;  
  dpi_Passport?: string;  
  email?: string;  
  phone?: string;  
  description?: string;  
  photoprrophy?: string;  
  state?: "active" | "inactive" | "pending";  
  type?: "cash" | "in-kind";  
  visible?: boolean;  
}
```

- Define la estructura para actualizar una donación.
- Todos los campos son opcionales para permitir actualizaciones parciales.

```
export interface UpdateDonationResponse {  
  message: string;  
  id: string;  
}
```

- Similar a `CreateDonationResponse`, pero en este caso indica que se ha actualizado una donación.

```
export interface DeleteDonationResponse {  
  message: string;  
}
```

- Confirma la eliminación de una donación con un mensaje de éxito.

Continuación Apéndice 1.

3. Componentes y funciones importantes

3.1 Modal Personalizado

Este componente muestra un modal que puede ser de éxito o error.

```
import React from 'react';
import { Modal, Typography, Button, Sheet } from '@mui/joy';
import CheckCircleIcon from '@mui/icons-material/CheckCircle';
import ErrorIcon from '@mui/icons-material/Error';

interface CustomModalProps {
  open: boolean;
  onClose: () => void;
  type: 'success' | 'error';
  title: string;
  message: string;
  onOkClick?: () => void;
}

const CustomModal: React.FC<CustomModalProps> = ({ open, onClose, type, title, message, onOkClick }) => {
  return (
    <Modal open={open} onClose={onClose}>
      <Sheet>
        {type === 'success' ? <CheckCircleIcon /> : <ErrorIcon />}
        <Typography>{title}</Typography>
        <Typography>{message}</Typography>
        <Button onClick={onOkClick || onClose}>Ok</Button>
      </Sheet>
    </Modal>
  );
};

export default CustomModal;
```

- Recibe propiedades como `type`, `title`, `message` y `onOkClick`.
- Muestra un icono de éxito o error según el tipo de mensaje.
- Permite cerrar el modal con un botón de `Ok`.

Continuación Apéndice 1.

3.2 Lista de Órdenes (OrderList)

```
import * as React from 'react';
import Box from '@mui/joy/Box';
import List from '@mui/joy/List';
import ListItem from '@mui/joy/ListItem';
import Typography from '@mui/joy/Typography';

const OrderList: React.FC<OrderListProps> = ({ rows, columns }) => {
  return (
    <Box>
      {rows.map((item) => (
        <List key={item.id}>
          <ListItem>
            {columns.map((column) => (
              <Typography key={column.id}>{column.renderCell ? column.renderCell(item) : item[column.id]}<
            ))}
          </ListItem>
        </List>
      ))}
    </Box>
  );
};

export default OrderList;
```

- Renderiza una lista de órdenes en formato de lista en dispositivos móviles.
- Usa `Typography` para mostrar los datos de cada columna.

Continuación Apéndice 1.

3.3 Tabla de Órdenes (OrderTable)

```
import * as React from 'react';
import Box from '@mui/joy/Box';
import Table from '@mui/joy/Table';
import Checkbox from '@mui/joy/Checkbox';

const OrderTable: React.FC<OrderTableProps> = ({ columns, rows }) => {
  return (
    <Table>
      <thead>
        <tr>
          <th>
            <Checkbox />
          </th>
          {columns.map((column) => (
            <th key={column.id}>{column.label}</th>
          ))}
        </tr>
      </thead>
      <tbody>
        {rows.map((row) => (
          <tr key={row.id}>
            <td><Checkbox /></td>
            {columns.map((column) => (
              <td key={column.id}>{column.renderCell ? column.renderCell(row) : row[column.id]}</td>
            ))}
          </tr>
        ))}
      </tbody>
    </Table>
  );
};

export default OrderTable;
```

- Se utiliza en pantallas grandes para mostrar datos tabulares.
- Incluye selección múltiple con `Checkbox`.

Continuación Apéndice 1.

3.4 Barra de Navegación (Barra)

```
import * as React from 'react';
import Drawer from '@mui/joy/Drawer';
import List from '@mui/joy/List';
import { NavLink, useNavigate } from 'react-router-dom';
import ListItemButton from '@mui/joy/ListItemButton';
import LogoutIcon from '@mui/icons-material/Logout';
import { routesPublicNavigation, getPrivateRoutesByRole } from '../constants';
import { useAuth } from '../../AuthProvider/useAuth';

interface BarraProps {
  open: boolean;
  setOpen: (open: boolean) => void;
}

const Barra: React.FC<BarraProps> = ({ open, setOpen }) => {
  const navigate = useNavigate();
  const { isLoggedIn, logout, user } = useAuth();
  const navigationRoutes = isLoggedIn ? getPrivateRoutesByRole(user?.role ?? "admin") : routesPublicNavigation;

  const handleLogout = () => {
    logout();
    navigate('/');
    setOpen(false);
  };

  return (
    <Drawer
      open={open}
      onClose={() => setOpen(false)}
      sx={{
        width: 250,
        flexShrink: 0,
        '& .MuiDrawer-paper': {
          width: 250,
          boxSizing: 'border-box',
        },
      }}
    >
      <NavLink to={route.path} style={{ textDecoration: 'none', color: 'inherit', width: '100%', text }}
        {route.label}
      </NavLink>
      </ListItemButton>
    </List>
  </Drawer>
);

export default Barra;
```

- Renderiza una barra lateral (Drawer) con navegación.
- Permite el cierre automático al seleccionar una opción.
- Incluye la opción de logout si el usuario está autenticado.

Continuación Apéndice 1.

3.5 Resumen del `AuthProvider` y `useAuth` en React

🔥 Descripción general

Este archivo define un **contexto de autenticación** (`AuthContext`) en React que maneja la autenticación de usuarios, incluyendo el inicio de sesión, la carga del perfil del usuario y el cierre de sesión. Utiliza `React Context` para proporcionar los datos y funciones relacionadas con la autenticación en toda la aplicación.

📄 Definición de interfaces

`AuthContextProps`

Define la estructura del contexto de autenticación:

- `user: UserProfile | null`: Contiene los datos del usuario autenticado o `null` si no ha iniciado sesión.
- `login(data: { email: string; password: string }): Promise<void>`: Método para iniciar sesión con un correo y contraseña.
- `loadUserProfile(): Promise<void>`: Método para cargar los datos del usuario autenticado.
- `isLoggedIn: boolean`: Indica si el usuario ha iniciado sesión.
- `logout(): void`: Método para cerrar sesión.

`AuthProviderProps`

Define las propiedades del proveedor de autenticación:

- `children: ReactNode`: Componentes hijos que estarán dentro del `AuthProvider`.

🌐 Creación del `AuthContext`

Se crea el contexto `AuthContext` con un estado inicial (`initialContextState`) que contiene valores predeterminados:

```
export const AuthContext = createContext<AuthContextProps>(initialContextState);
```

Esto permite que cualquier componente dentro del `AuthProvider` pueda acceder a los datos y métodos de autenticación.

🔧 Implementación de `AuthProvider`

El `AuthProvider` es el componente que maneja la autenticación en la aplicación. Su estructura es la siguiente:

🔥 Estados internos

- `loading`: Indica si se está realizando una operación de autenticación (ej. inicio de sesión).
- `user`: Almacena la información del usuario autenticado.
- `isLoggedIn`: Controla si el usuario está autenticado.
- `errorModalOpen`: Estado para manejar si el modal de error debe mostrarse.
- `errorMessage`: Mensaje de error a mostrar en el modal.
- `titleError`: Título del modal de error.

Continuación Apéndice 1.

🔴 Métodos principales

• login

```
const login = async (data: { email: string; password: string }) => {
  setLoading(true);
  try {
    await loginService(data);
    await loadUserProfile();
  } catch (error) {
    setTitleError("Credenciales Inválidas");
    setErrorMessage("Error al iniciar sesión. Verifique sus credenciales y vuelva a intentarlo.");
    setErrorModalOpen(true);
  } finally {
    setLoading(false);
  }
};
```

- Llama al servicio de inicio de sesión.
- Si tiene éxito, carga el perfil del usuario.
- Si falla, muestra un mensaje de error en un `CustomModal`.

• logout

```
const logout = () => {
  Cookies.remove('auth_token');
  setUser(null);
  setIsLoggedIn(false);
};
```

- Elimina la cookie de autenticación.
- Resetea el estado del usuario y la sesión.

• loadUserProfile

```
const loadUserProfile = async () => {
  try {
    const profile = await getUserProfile();
    setUser(profile);
    setIsLoggedIn(true);
  } catch (error) {
    console.error('Failed to load user profile:', error);
    setIsLoggedIn(false);
  }
};
```

- Obtiene la información del usuario autenticado.
- Si hay un error, marca la sesión como no iniciada.

🔵 Hook personalizado useAuth

Se crea un hook para consumir el contexto de autenticación:

```
export const useAuth = (): AuthContextProps => {
  const context = useContext(AuthContext);
  if (!context) throw new Error('useAuth must be used within an AuthProvider');
  return context;
};
```


Continuación Apéndice 1.

- Verifica si el contexto existe.
- Si no se encuentra dentro del `AuthProvider`, lanza un error.

Renderización del `AuthProvider`

```
return (  
  <AuthContext.Provider value={{ user, login, logout, loadUserProfile, isLoggedIn }}>  
    {loading && (  
      <Box sx={{  
        position: "absolute",  
        top: 0,  
        left: 0,  
        width: "100%",  
        height: "100%",  
        display: "flex",  
        justifyContent: "center",  
        alignItems: "center",  
        backgroundColor: "rgba(255, 255, 255, 0.7)",  
        zIndex: 10,  
      }}>  
        <CircularProgress sx={{ "--CircularProgress-size": "80px" }} />  
      </Box>  
    )}  
    {children}  
    <CustomModal open={errorModalOpen} onClose={() => setErrorModalOpen(false)} type="error" title={titleE}>  
      </AuthContext.Provider>  
    )  
  );
```

- Muestra un `CircularProgress` cuando `loading` es `true`.
- Renderiza el `CustomModal` en caso de error.
- Proporciona el contexto de autenticación a los componentes hijos.

Conclusión

Este archivo define un sistema de autenticación basado en `React Context`, que permite:

- ✓ Iniciar sesión y almacenar los datos del usuario.
- ✓ Cargar la información del usuario autenticado.
- ✓ Manejar errores de autenticación con un modal personalizado.
- ✓ Implementar un hook `useAuth` para acceder al contexto en cualquier componente.
- ✓ Mostrar un `loading spinner` mientras se realiza una autenticación.

Este enfoque modular permite gestionar la autenticación de manera eficiente en una aplicación de React.

Este apartado de **Funciones Generales** explica en detalle cada implementación clave, proporcionando ejemplos de código completos para una mejor comprensión del sistema.

Continuación Apéndice 1.

Implementación de `getUserProfile`

Definición de la función

```
export const getUserProfile = async (): Promise<UserProfile> => {
  try {
    // Obtener el token de autenticación desde las cookies
    const token = Cookies.get('auth_token');
    console.log(token);

    // Si no se encuentra el token, lanzar un error
    if (!token) throw new Error('Token not found');

    // Realizar una solicitud GET al backend con el token en el encabezado
    const response: AxiosResponse<UserProfile> = await axios.get(`${BASE_URL}/user/info`, {
      headers: {
        Authorization: `${token}`,
        'Content-Type': 'application/json',
      },
      withCredentials: true,
    });

    // Devolver los datos del usuario obtenidos del servidor
    return response.data;
  } catch (error) {
    console.error('Error fetching user profile:', error);
    throw new Error('Error fetching user profile');
  }
};
```

Pasos detallados:

1. Obtener el token desde las cookies:

- Se usa `Cookies.get('auth_token')` para recuperar el token almacenado.
- Si no hay token, se lanza un error.

2. Realizar la solicitud HTTP:

- Se usa `axios.get` para obtener los datos del usuario.
- Se pasa el token en el header `Authorization`.
- Se define el `Content-Type` como `application/json`.
- Se habilita `withCredentials: true` para que las cookies sean enviadas con la petición.

3. Retornar los datos del usuario:

- Si la solicitud es exitosa, se devuelve `response.data`.
- Si hay un error, se captura y se lanza un nuevo error con un mensaje genérico.

Manejo de errores

- Si no hay token, se lanza un error `Token not found`.
- Si ocurre un problema en la solicitud, se captura el error y se imprime en consola.
- Se lanza un nuevo error `Error fetching user profile` para informar sobre el fallo.

Continuación Apéndice 1.

3.6 Resumen de `getUserProfile` en React

Descripción general

Esta función, `getUserProfile`, es responsable de obtener el perfil de usuario desde el backend utilizando un **token almacenado en una cookie**. Se basa en **Axios** para realizar la solicitud HTTP y utiliza la variable de entorno `VITE_BACK_URL` como base para la URL de la API.

Dependencias utilizadas

- `axios`: Para realizar solicitudes HTTP al backend.
- `js-cookie`: Para obtener el token de autenticación almacenado en cookies.
- `UserProfile`: Una interfaz que define la estructura del perfil de usuario.

Configuración de la URL base

La función obtiene la URL base del backend desde una variable de entorno:

```
const BASE_URL = import.meta.env.VITE_BACK_URL || '';  
console.log('Backend URL:', BASE_URL);
```

- Se utiliza `import.meta.env.VITE_BACK_URL` para leer la URL del backend desde las variables de entorno.
- Si no se encuentra definida, se asigna un string vacío (`''`).
- Se imprime en consola la URL para depuración.

Continuación Apéndice 1.

Despliegue de la Aplicación React a CloudFront

Este apartado explica cada paso del proceso de despliegue de una aplicación React en AWS CloudFront utilizando GitHub Actions.

YAML de Despliegue

```
name: Build and Deploy React App to CloudFront

on:
  push:
    branches: [developTest]

jobs:
  build-and-deploy:
    name: Build and Deploy
    runs-on: ubuntu-latest
    env:
      BUCKET: jepa-guatemala
      DIST: dist
      REGION: us-east-1
      DIST_ID: E3TN9YWB479FY

    steps:
      - name: Checkout
        uses: actions/checkout@v3

      - name: Configure AWS Credentials
        uses: aws-actions/configure-aws-credentials@v1
        with:
          aws-access-key-id: ${ secrets.AWS_ACCESS_KEY_ID }
          aws-secret-access-key: ${ secrets.AWS_SECRET_ACCESS_KEY }
          aws-region: ${ env.REGION }

      - name: Clean npm cache and remove node_modules
        run: |
          npm cache clean --force
          rm -rf node_modules package-lock.json

      - name: Install dependencies
        run: |
          node --version
          npm i --legacy-peer-deps

      - name: Build Static website
        env:
          VITE_BACK_URL: https://azg8de6i2d.execute-api.us-east-1.amazonaws.com/dev
          VITE_LINK_S3: https://images-jepa-guatemala.s3.us-east-1.amazonaws.com/
          VITE_JWT_SECRET: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
        run: |
          npm run build

      - name: Sync files to S3 bucket
        run: |
          aws s3 sync --delete ${ env.DIST } s3://${ env.BUCKET }

      - name: Invalidate CloudFront Cache
        run: |
          aws cloudfront create-invalidation --distribution-id ${ env.DIST_ID } --paths "/*"
```

Continuación Apéndice 1.

Explicación Paso a Paso

1. Activación del Workflow

```
on:  
  push:  
    branches: [developTest]
```

- Este workflow se ejecuta automáticamente cuando hay un `push` a la rama `developTest`.

2. Configuración del Job

```
jobs:  
  build-and-deploy:  
    name: Build and Deploy  
    runs-on: ubuntu-latest
```

- Define un job llamado `build-and-deploy` que se ejecuta en `ubuntu-latest`.

3. Variables de Entorno

```
env:  
  BUCKET: jepa-guatemala  
  DIST: dist  
  REGION: us-east-1  
  DIST_ID: E3TN9YWM479FY
```

- `BUCKET`: Nombre del bucket S3 donde se subirá la aplicación.
- `DIST`: Directorio donde se genera el build.
- `REGION`: Región de AWS donde están los servicios.
- `DIST_ID`: ID de distribución de CloudFront.

4. Clonación del Repositorio

```
- name: Checkout  
  uses: actions/checkout@v3
```

- Descarga el código fuente del repositorio.

Continuación Apéndice 1.

5. Configuración de AWS Credentials

```
- name: Configure AWS Credentials
  uses: aws-actions/configure-aws-credentials@v1
  with:
    aws-access-key-id: ${ secrets.AWS_ACCESS_KEY_ID }
    aws-secret-access-key: ${ secrets.AWS_SECRET_ACCESS_KEY }
    aws-region: ${ env.REGION }
```

- Configura las credenciales de AWS utilizando secretos almacenados en GitHub Actions.

6. Limpieza de Caché y Dependencias

```
- name: Clean npm cache and remove node_modules
  run: |
    npm cache clean --force
    rm -rf node_modules package-lock.json
```

- Borra la caché de `npm` y elimina `node_modules` para evitar conflictos.

7. Instalación de Dependencias

```
- name: Install dependencies
  run: |
    node --version
    npm i --legacy-peer-deps
```

- Instala las dependencias de `npm`.

8. Construcción del Proyecto

```
- name: Build Static website
  env:
    VITE_BACK_URL: https://azg0de6i2d.execute-api.us-east-1.amazonaws.com/dev
    VITE_LINK_S3: https://images-jepa-guatemala.s3.us-east-1.amazonaws.com/
    VITE_JWT_SECRET: ...
  run: |
    npm run build
```

- Ejecuta el comando `npm run build` para generar los archivos estáticos del sitio web.

Continuación Apéndice 1.

9. Subida de Archivos a S3

```
- name: Sync files to S3 bucket
run: |
  aws s3 sync --delete ${ env.DIST } s3://${ env.BUCKET }
```

- Sincroniza los archivos generados en `dist` con el bucket de S3.
- Usa `--delete` para eliminar archivos antiguos.

10. Invalidación de Caché en CloudFront

```
- name: Invalidate CloudFront Cache
run: |
  aws cloudfront create-invalidation --distribution-id ${ env.DIST_ID } --paths "/*"
```

- Fuerza la actualización de archivos en CloudFront eliminando la caché.

Este apartado de **Despliegue de la Aplicación React a CloudFront** explica cada paso del proceso para que el despliegue sea claro y comprensible.

Conclusión

Este manual cubre el **frontend** de **jepaBack**, utilizado tanto por **Asocoprode** como **Jepa Guatemala**. Con la implementación de **React**, **React Router** y **MUI Joy**, se garantiza una experiencia de usuario optimizada. Además, la implementación de despliegue automático en **AWS CloudFront** facilita el mantenimiento y las actualizaciones de la aplicación.

Nota. Elaboración propia, realizado con