Arévalo Information Technology Company, S.A. de C.V. (AIT Company S.A. de C.V.)

Servicios de desarrollo de software para conexión del sistema de información de sanidad agropecuaria (SISA) con sistema de notificaciones MSF.

PROPUESTA TÉCNICA

Referencia No: IESC ATRACSI-RFP-SV02-2024.

Contenido

1 INTRODUCCIÓN

En el contexto de la Unión Aduanera y la integración económica en Centroamérica, la empresa AIT se complace en presentar una propuesta técnica de servicios de desarrollo de software para integrar de manera eficiente el Sistema de Información de Sanidad Agropecuaria (SISA) del Ministerio de Agricultura y Ganadería (MAG) con el Sistema de Notificaciones MSF de la Secretaría de Integración Económica Centroamericana (SIECA).

Como parte integral de esta propuesta, la empresa AIT ha elaborado un plan de trabajo detallado (Preliminar) que describe cómo se llevará a cabo el proyecto de integración entre el SISA y el MSF. El plan se desarrolla utilizando el marco de trabajo Ágil SCRUM, este plan incluye épicas como hitos del desarrollo, las historias de usuario que con los RFP se pueden definir y las tareas asociadas a dichas historias de usuario, también, se han definido los sprints en los que se desglosará el trabajo del proyecto. Además, se contempla la asignación de personal técnico altamente capacitado y especializado en el desarrollo de software y en la integración de sistemas complejos.

También, Presentamos el enfoque técnico con el cual pretendemos abordar las soluciones a las necesidades planteadas en los RFP del proyecto.

El equipo técnico de AIT cuenta con la experiencia y la expertise necesarias para afrontar los desafíos que implica la integración de sistemas de información tan críticos como el SISA y el MSF. Con un enfoque centrado en la calidad, la eficiencia y la innovación, nuestro equipo se compromete a garantizar el éxito del proyecto y a cumplir con los objetivos establecidos en colaboración con el Ministerio de Agricultura y Ganadería y la Secretaría de Integración Económica Centroamericana.

2 ENFOQUE TECNICO Y GESTIÓN.

2.1 DISEÑO DE LA SOLUCION TÉCNICA.

Vamos a abordar diferentes enfoques para las diferentes necesidades que se desean solventar, antes que realizar la propuesta, debemos de conocer un poco de contexto sobre los módulos adscritos a SISA actualmente,

2.1.1 Contexto sobre arquitectura y patrones de diseño.

En el contexto de los proyectos Java Enterprise Archive (EAR), se sigue principalmente la arquitectura J2EE (Java 2 Platform, Enterprise Edition). Esta arquitectura está diseñada para desarrollar aplicaciones empresariales de gran escala. La estructura propuesta para el proyecto EAR en SISA MAG y los patrones de diseño son los siguientes:

Arquitectura: J2EE, Multicapa:

- Capa de presentación: Contiene componentes como servlets, JavaServer Pages (JSP), JavaServer Faces (JSF), que manejan la interfaz de usuario y la interacción con el cliente, para nuestro caso en particular, se utilizar el framework Primefaces integrado con JSF.
- Capa de negocio: Implementa la lógica de negocio utilizando Enterprise JavaBeans (EJB), que pueden ser de diferentes tipos (session beans, entity beans, message-driven beans), para nuestro caso en particular entity beans para el mapeo objeto relacional con hibérnate.
- Capa de persistencia: Maneja el almacenamiento y recuperación de datos, utilizando tecnologías como Java Persistence API (JPA) o JDBC, vamos a utilizar JPA Sobre Hibernate.
- Capa de integración: Facilita la interacción con otros sistemas y servicios, a menudo a través de APIs, servicios web (SOAP/REST), o mensajería (JMS), para el caso que nos atañe, utilizaremos REST Web Service para exponer servicios y Clientes específicos en cada módulo web para los clientes que consumen.
- Componentes y Módulos:
    - EAR (Enterprise Archive): Es el archivo contenedor principal que puede incluir módulos EJB (archivos JAR), módulos web (archivos WAR), y otros recursos necesarios para la aplicación.
    - WAR (Web Archive): Contiene la capa de presentación (servlets, JSP, HTML, etc.).
    - JAR (Java Archive): Contiene la lógica de negocio (EJBs, bibliotecas de utilidades, etc.).

2.1.2 Modificaciones a SISA, propuesta de arquitectura.

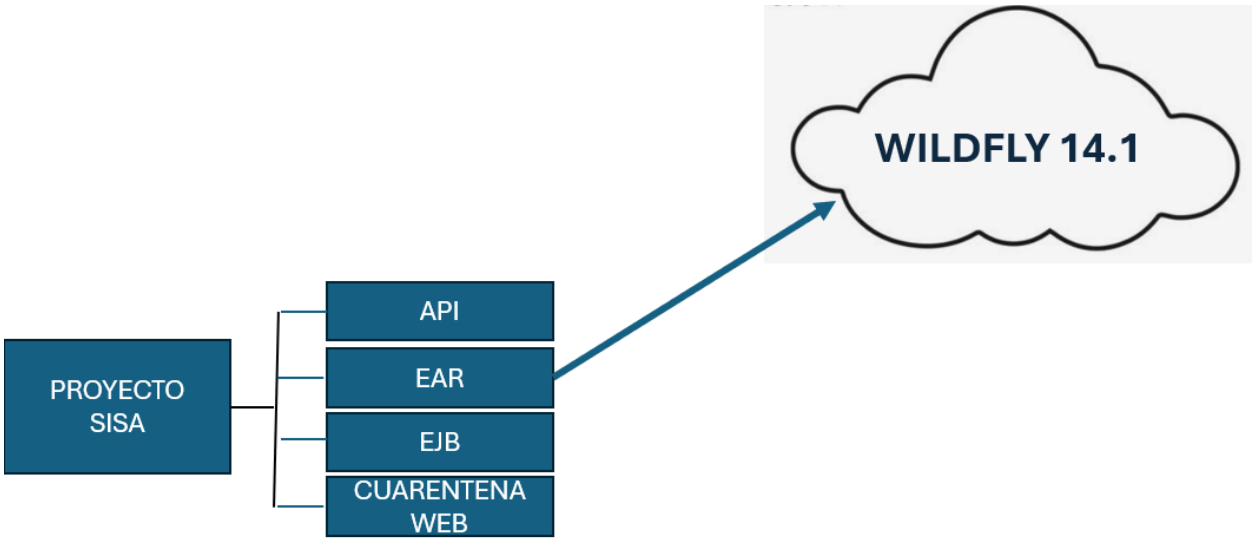
Desglose de capas de la arquitectura:

Vamos a aprovechar, los desarrollos con los que el MAG ya cuenta y vamos a agregar las funcionalidades requeridas.
El proyecto SISA del MAG, está compuesto por varias capas, de las cuales podemos mencionar:

- SISA-EAR: Capa que contiene el disparador para el empaquetado completo de las demás capas, es la capa común de comunicación entre las demás capas, en esta capa, en la carpeta target se construirá el empaquetado que posteriormente se subirá al servidor de aplicaciones.
- SISA-EJB. Es la que contiene la comunicación directa a las bases de datos de sisa normativo.
- SISA-API. Expone servicios web que pueden ser utilizado por otras instituciones interesadas en consumir información que proviene del MAG y pase a través del SISA Normativo.
- CITES-WEB. Contiene módulos Web para el control de especies en peligro de extinción.
- REGISRO-WEB. Registro web contiene módulos para el control de registro y fiscalización.

- CUARENTENA-WEB. En Cuarentena Web se encuentra la administración de los productos zoo y fitosanitarios tanto de importación como exportación, estos productos se encuentran ya actualizados en código arancelario enmienda 7, adicionalmente podemos encontrar las reglas de ingreso al país, los requerimientos Fito y zoo sanitarios por producto-país, ente otros.

De esas capas, para el entregable, ***ajustes al módulo normativo SISA***, vamos a utilizar las capas SISA-EAR, SISA-EJB, SISA-API y CUARENTENA-WEB, todo corriendo bajo el servidor de aplicaciones Wildfly 14.1.



Patrones de Diseño propuestos, modificaciones a SISA.

- **Patrón MVC (Model-View-Controller)**
  Se utiliza comúnmente en la capa de presentación. Primfeaces sobre JSF y frameworks como Spring MVC aplican este patrón para separar la lógica de negocio, la lógica de presentación y la gestión de eventos.
- **Patrón DAO (Data Access Object)**
  Utilizado en la capa de persistencia para abstraer y encapsular el acceso a la base de datos, proporcionando una interfaz específica para cada entidad.
- **Patrón DTO (Data Transfer Object)**
  Utilizado para transferir datos entre diferentes capas de la aplicación o consumir o exponer datos de servicios específicos.
- **Patrón Facade**
  Proporciona una interfaz unificada para un conjunto de interfaces en un subsistema. Se utiliza para simplificar las interacciones con subsistemas complejos.

Estos patrones de diseño ayudan a crear aplicaciones que son mantenibles, escalables y fáciles de entender, al tiempo que aprovechan al máximo las capacidades de la plataforma J2EE, todos serán utilizados en el desarrollo de las modificaciones solicitadas.

2.1.3 Sistema de colas e interconexiones, propuestas

2.1.3.1 Propuesta 1: RabbitMQ.

Utilizar RabbitMQ como intermediario para la gestión de las colas.

RabbitMQ es un sistema de mensajería que sigue el modelo tradicional de broker de mensajes basado en colas. La lógica detrás de RabbitMQ incluye:

1. Modelo de Cola y Enrutamiento:
   - Productores: Envían mensajes a un exchange.
   - Exchanges: Reciben mensajes de los productores y los encaminan a una o más colas basadas en reglas de enrutamiento. Hay varios tipos de exchanges como direct, topic, fanout, y headers.
   - Colas: Almacenan mensajes hasta que los consumidores los procesan.
   - Consumidores: Recuperan mensajes de las colas y los procesan.
2. Tipos de Exchanges:
   - Direct Exchange: Enruta mensajes a colas con una clave de enrutamiento exacta.
   - Topic Exchange: Enruta mensajes a una o más colas basándose en patrones de coincidencia de claves.
   - Fanout Exchange: Enruta mensajes a todas las colas vinculadas sin considerar ninguna clave de enrutamiento.
   - Headers Exchange: Enruta mensajes basados en coincidencias de cabeceras.
3. Mensajes y Ack:
   - Los mensajes pueden ser persistentes o no persistentes.
   - Los consumidores deben confirmar (acknowledge) la recepción de mensajes para que RabbitMQ los considere procesados. Si no se reconoce, el mensaje puede reenviarse a otro consumidor.
4. Características Adicionales:
   - Soporte para múltiples protocolos (AMQP, MQTT, STOMP).
   - Implementación de patrones como RPC (Remote Procedure Call) y colas de trabajo.

2.1.3.2 Propuesta 2: Apache Kafka .

Apache Kafka es una plataforma de streaming de datos distribuida que sigue un modelo de registro de commit distribuido. La lógica detrás de Kafka incluye:

1. Modelo de Registro:
   - Productores: Publican mensajes a un topic.
   - Topics: Divididos en particiones, donde cada partición es una secuencia ordenada de registros (mensajes).
   - Consumidores: Suscriben a topics y leen mensajes en el orden en que se almacenaron dentro de las particiones.
2. Almacenamiento y Retención:
   - Los mensajes en Kafka se almacenan en el disco y se mantienen durante un período configurado (basado en tiempo o tamaño).
   - Esto permite a los consumidores volver a leer mensajes históricos, lo que es útil para procesamiento de datos y análisis.
3. Escalabilidad y Particiones:
   - Cada topic puede tener múltiples particiones, lo que permite distribuir la carga de trabajo entre varios nodos y consumidores.
   - Los consumidores en un grupo de consumidores se distribuyen las particiones, lo que permite un procesamiento paralelo eficiente.
4. Durabilidad y Replicación:
   - Kafka replica datos entre múltiples nodos para asegurar durabilidad y alta disponibilidad.
   - Cada partición tiene un líder y varios seguidores; los productores y consumidores interactúan principalmente con los líderes.
5. Offset y Estado del Consumidor:
   - Los consumidores mantienen un registro de su posición (offset) dentro de cada partición.

- Los offsets permiten a los consumidores procesar mensajes exactamente una vez o al menos una vez, dependiendo de la configuración.

2.1.3.3 Propuesta 3: Message Broker a medida.

Creación de un sistema de colas (broker) a medida de las necesidades del MAG.
Para crear nuestro propio bróker:
Definición de Requisitos. Aspectos clave a considerar:

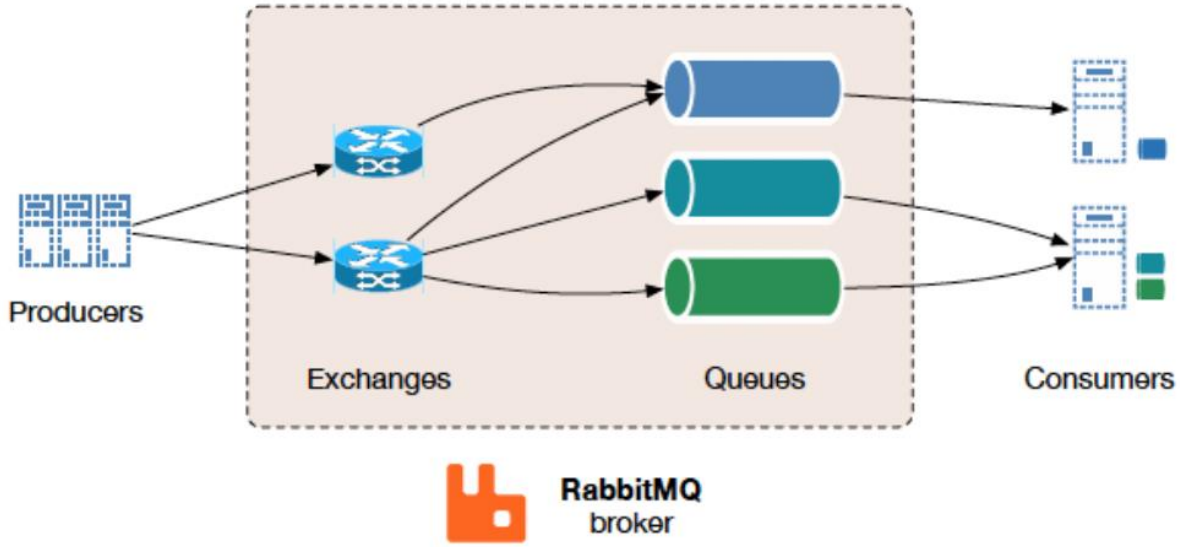
- Protocolos de comunicación: ¿Qué protocolos soportará el broker? (HTTP, TCP, WebSockets, etc)
- Persistencia: ¿Cómo se almacenarán los mensajes? ¿Serán persistentes o solo en memoria?, para el tema del monitoreo puede ser en base de datos.
- Escalabilidad: ¿El broker necesita soportar múltiples instancias y balanceo de carga?
- Seguridad: ¿Manejar la autenticación y la autorización con JWT?
- Características adicionales: ¿Soportará enrutamiento, filtros, transformaciones de mensajes, etc.?

Diseño de la Arquitectura. Componentes principales:

- Cliente: La interfaz que los productores y consumidores usan para enviar y recibir mensajes.
- Servidor: El componente central que recibe, almacena y distribuye mensajes.
- Colas/Topics: Estructuras de datos para gestionar mensajes.
- Persistencia: Mecanismo para almacenar mensajes de forma persistente.

2.1.4 Propuesta de arquitectura sugerida.

En base a las propuestas anteriores, sugerimos utilizar RabbitMQ, es un broker de mensajería de código abierto, distribuido y escalable, que sirve como intermediario para la comunicación eficiente entre productores y consumidores.



2.1.5 Propuesta de sistema monitor de colas.

Proponemos usar el **RabbitMQ Management Plugin** y una biblioteca HTTP (**org.apache.httpcomponents**) para hacer solicitudes a la API de gestión de RabbitMQ.

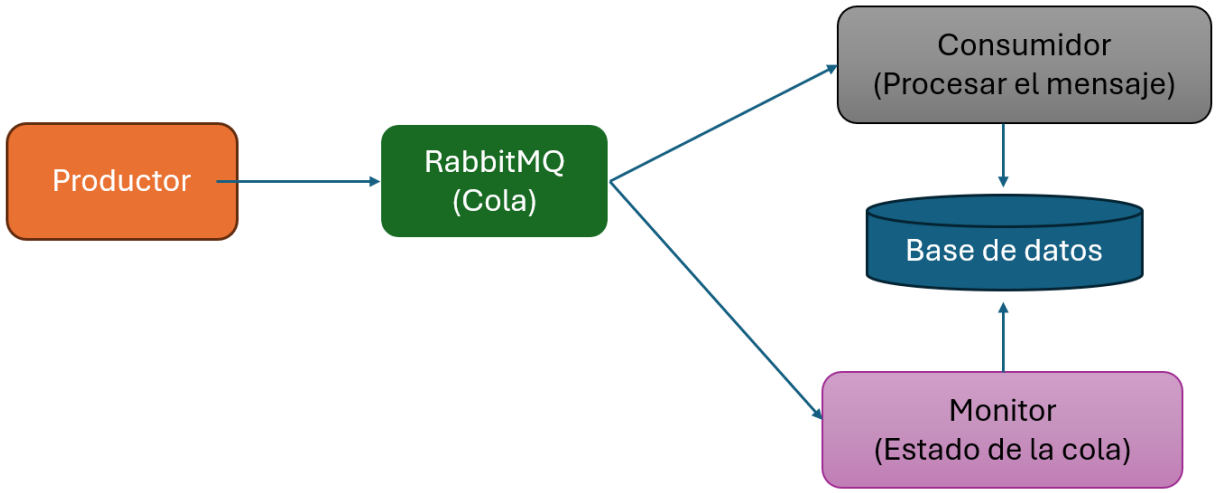


Diagrama del sistema que muestra la interacción entre RabbitMQ, el productor, el consumidor y el monitoreo de las colas:

- Productor (Producer): Envía mensajes a la cola de RabbitMQ.
- RabbitMQ (Cola): Recibe los mensajes enviados por el productor., Almacena los mensajes hasta que el consumidor los procese.
- Consumidor: Recibe mensajes de la cola de RabbitMQ, Procesa los mensajes y los guarda en la base de datos.
- Base de Datos: Almacena los mensajes procesados por el consumidor.
- Monitor: Consulta el estado de la cola de RabbitMQ y guarda esta información en la base de datos.

2.1.6 Propuesta técnica sobre el control de cambios del código durante el desarrollo y políticas:

Utilizar gitlab como gestor de repositorios de código.
Propuestas de ramas en sisa.

Rama	Uso	Política de cambios
Master	Contendrá los cambios que suben a producción.	Se debe solicitar autorización a DTIT MAG para hacer confirmación a esta rama.
QA	Rama que contiene los cambios que son publicados a la rama de QA.	Se debe solicitar autorización a DTIT MAG para hacer merge a esta rama. Se debe de comunicar efectivamente cuando se realizará una cambio a rama QA para asegurar de no interrumpir el trabajo de otras consolatorias.
Ralease	Es la rama previa a la subida a producción, siempre nacerá de la rama master, es donde se hacen las pruebas finales previo al merge con master.	Siempre se debe de crear a partir de la rama master. Debe de mandar a hacer merge desde la rama feature/x.

		Debe pasar primero por la rama QA y sus respectivas pruebas para luego pasar a ser reléase.
Feature/x	Son las ramas de los desarrolladores, donde trabajan en ambiente locales y de servidores de desarrollo	X hace referencia a que debe llevar el estándar de nombre de ramas que sea definidor. El estándar definido será: Feature/[Sprint,taks], ejemplo: Feature-sprint1-task10

2.1.7 Estrategia de despliegue para publicar en los diferentes ambientes.

En ambiente de desarrollo:

- Pruebas en ambientes locales.
- Empaquetado de proyecto.
- Despliegue en ambiente de desarrollo.

Ambiente de QA.

- Comunicación con equipos de DTIT para anunciar el paso a QA.
- Merge de rama Feature a rama QA.
- Crear ticket para el paso QA con infraestructura.
- Despliegue en ambiente de QA.

Ambiente de Producción.

- Crear rama reléase a partir de máster.
- Merge de rama Feature a rama reléase.
- Despliegue de rama reléase en QA.
- Ejecución de pruebas en QA.
- Si todo sale según lo planificado.
- Despliegue en producción.
- Si todo sale bien, merge a rama máster.
- Si ocurre algún inconveniente, rollback a rama master.

2.1.8 Tecnologías a utilizar:

Categoría	Tecnología
IDE	Eclipse Visual Studio code
Lenguajes de Programación	Java PLSQL
Frameworks y Librerías	JSF Primefaces Spring Hibernate EJB Sonar Qube

Tecnologías de Web Services	JAX-RPC JAX-WS Restful SOAP WSDL XML JSON
Servidores de Aplicación	Servidor de aplicaciones WildFly Servidor de aplicaciones Jboss EAP applications
Bases de datos	Bases de datos Oracle
Manejo de colas (Borkers)	Rabbit MQ
Control de versiones	GIT
Control de dependencias	Maven

2.2 METODOLOGÍA DE GESTION.

2.2.1 Propuesta de marco de trabajo para la gestión: SCRUM.

El Desarrollo de Software Ágil es un tipo de metodología de desarrollo que anticipa la necesidad de flexibilidad y aplica un nivel de practicidad a la entrega del producto terminado y Scrum es un marco de gestión de proyectos de metodología ágil que ayuda a los equipos a estructurar y gestionar el trabajo mediante un conjunto de valores, principios y prácticas estándares que provee dicho marco de trabajo.
Ventajas de la metodología propuesta:

- Mejora la calidad alertando y reduciendo los errores o problemas.
- Mejora satisfacción del cliente por las entregas continuas.
- Mayor compromiso, motivación e implicación.
- Mayor rapidez y eficiencia, y menores costes.
- Aumento de la productividad.
- Acelera el retorno de inversión.

2.2.1.1 Implementación de la metodología.

Vamos a utilizar las epics, issues, task y sprints para organizar el trabajo.
Las epics, issues y tasks son tipos de elementos de trabajo que se utilizan para organizar y gestionar el trabajo en un proyecto de desarrollo de software. Cada uno tiene un propósito y un nivel de granularidad diferente

A continuación, nos familiarizamos con algunos términos recurrentes en un marco de trabajo como SCRUM.

1. Epic:
   - Definición: Una epic es una gran unidad de trabajo que puede abarcar múltiples sprints o versiones. Representa una funcionalidad o característica significativa del producto.
   - Uso: Se utiliza para agrupar un conjunto de features o user stories relacionadas que juntas cumplen un objetivo mayor.

- Ejemplo: "Implementar sistema de autenticación de usuarios" podría ser una epic que incluya múltiples user stories como "Permitir el inicio de sesión con correo electrónico y contraseña", "Implementar recuperación de contraseña", etc.

2. Issue:
   - Definición: Un issue generalmente se refiere a una historia de usuario, problema, error o bug que necesita ser resuelto. Aunque en algunos casos puede utilizarse de manera más amplia para representar cualquier tipo de trabajo que no encaje perfectamente en otras categorías.
   - Uso: Se utiliza para rastrear y gestionar historias de usuario, problemas o bugs que deben ser solventados en el sistema.
   - Ejemplo: "Error en la validación del formulario de registro" sería un issue que describe un problema específico que necesita ser solucionado.
3. Task:
   - Definición: Una task es una unidad de trabajo más pequeña y concreta. Es una acción específica que debe completarse como parte de una user story o issue.
   - Uso: Se utiliza para descomponer user stories o issues en acciones más pequeñas y manejables que pueden ser asignadas a diferentes miembros del equipo.
   - Ejemplo: Dentro de la user story "Permitir el inicio de sesión con correo electrónico y contraseña", podría haber tasks como "Diseñar la interfaz de usuario para el formulario de inicio de sesión", "Desarrollar la lógica de autenticación en el backend", "Escribir pruebas unitarias para la funcionalidad de inicio de sesión".
4. Sprint: Es un ciclo de trabajo de duración fija, durante el cual se desarrolla una parte del producto o proyecto. Es uno de los elementos fundamentales del marco de trabajo Scrum y tiene las siguientes características:
   - Duración Fija:
     Un sprint tiene una duración fija, generalmente de una a cuatro semanas. La duración se decide al inicio del proyecto y se mantiene constante para todos los sprints.
   - Objetivo Claro:
     Cada sprint tiene un objetivo específico, conocido como el "Sprint Goal", que define lo que se espera lograr al final del sprint.
   - Incremento de Producto:
     Al final de cada sprint, el equipo debe tener un incremento de producto potencialmente entregable, que es una versión funcional del producto que se ha mejorado con las nuevas funcionalidades desarrolladas durante el sprint.
   - Eventos en un Sprint:
     a. Sprint Planning (Planificación del Sprint): Reunión al inicio del sprint donde el equipo selecciona y planifica el trabajo que se completará durante el sprint.
     b. Daily Scrum (Scrum Diario): Reunión diaria de 15 minutos donde el equipo revisa el progreso y ajusta el plan para el día.
     c. Sprint Review (Revisión del Sprint): Reunión al final del sprint donde se presenta el incremento del producto al cliente y otras partes interesadas para obtener feedback.
     d. Sprint Retrospective (Retrospectiva del Sprint): Reunión al final del sprint donde el equipo reflexiona sobre el proceso y busca maneras de mejorar en el próximo sprint.
5. Sprint backlog.
   El Sprint Backlog es una lista de elementos seleccionados del Product Backlog que el equipo de desarrollo se compromete a completar durante el próximo Sprint. Esta lista incluye las historias de usuario, tareas y cualquier otro trabajo que el equipo considere necesario para alcanzar el objetivo del Sprint.

6. Un stakeholder: es cualquier individuo, grupo o entidad que tiene un interés o "stake" en un proyecto, organización o iniciativa, y que puede afectar o verse afectado por las acciones, decisiones, políticas o resultados relacionados con ese proyecto u organización.

El adoptar un ciclo de trabajo fijo a través del sprint no proveerá de:
Iteración y Adaptación: Los sprints permiten iterar sobre el producto, adaptándolo constantemente en función del feedback recibido.
Transparencia: Los eventos del sprint, como las revisiones y retrospectivas, promueven la transparencia y la inspección del progreso y los procesos.

Mejora Continua: Las retrospectivas permiten al equipo identificar áreas de mejora y aplicar cambios en el próximo sprint, fomentando la mejora continua.

2.2.2 Gestión del riesgo y control de cambios.

Etapa Scrum	Gestión de Riesgos	Control de Cambios
Sprint Planning:	Durante la planificación del sprint, el equipo identifica posibles riesgos asociados con las historias de usuario seleccionadas. Se discuten los riesgos y se planifican tareas para mitigarlos.	Durante la planificación del sprint, se decide qué elementos del Product Backlog se incluirán en el sprint actual. Una vez que el Sprint Backlog está establecido, se espera que permanezca estable durante el sprint, aunque se pueden hacer ajustes menores si son necesarios y acordados por el equipo.
Daily Scrum	Los miembros del equipo discuten diariamente su progreso y cualquier impedimento que hayan encontrado. Cualquier riesgo emergente puede ser identificado rápidamente y abordado de inmediato.	
Sprint Review	Durante la revisión del sprint, el equipo y los interesados pueden identificar nuevos riesgos basados en el trabajo completado y los resultados obtenidos.	
Sprint Retrospective	El equipo reflexiona sobre el sprint pasado y discute qué salió bien y qué no. Se identifican riesgos recurrentes y se buscan estrategias para mitigarlos en futuros sprints	El equipo revisa su proceso de trabajo y hace ajustes para mejorar en el próximo sprint. Esto puede incluir cambios en la forma en que se manejan las historias de usuario, los riesgos y otros aspectos del proyecto

COMUNICACIÓN:

La gestión efectiva de la comunicación en Scrum se centra en establecer canales claros, fomentar la transparencia, promover la colaboración activa y utilizar herramientas adecuadas para facilitar la interacción entre los miembros del equipo y otros stakeholders. Esto ayuda a mantener el equipo alineado con los objetivos del proyecto y a manejar eficazmente los cambios y desafíos que puedan surgir durante el desarrollo de software.
Prácticas y principios clave para gestionar la comunicación efectivamente propuestas para el proyecto:

1. Canales de Comunicación Transparentes: Es fundamental establecer canales de comunicación claros y accesibles para todos los miembros del equipo. Esto incluye reuniones regulares como las Daily Scrums, así como herramientas digitales para la comunicación síncrona y asíncrona.
2. Reuniones Estructuradas: Las reuniones prescritas por Scrum, como la Sprint Planning, la Daily Scrum, la Sprint Review y la Sprint Retrospective, son momentos clave para la comunicación. Estas deben ser bien estructuradas y enfocadas en los objetivos específicos de cada reunión.
3. Transparencia: Scrum promueve la transparencia en todos los aspectos del trabajo. Esto significa que la información relevante debe ser accesible para todos los miembros del equipo en todo momento. Las herramientas como azure boards utilizada en este proyecto, facilitan esta transparencia.
4. Colaboración Activa: La comunicación en Scrum no se limita a actualizar el progreso, sino que también implica una colaboración activa entre los miembros del equipo. Esto puede incluir la resolución conjunta de problemas, la planificación de tareas y la revisión de resultados.
5. Herramientas de Comunicación: Utilizar herramientas adecuadas para la comunicación es crucial. Esto puede incluir sistemas de gestión de proyectos como Jira, Project, herramientas de chat como Slack o Microsoft Teams, y plataformas para reuniones virtuales como Zoom o Google Meet.
6. Comunicación de Progreso y Obstáculos: Los miembros del equipo deben comunicar regularmente su progreso y cualquier obstáculo que pueda afectar la consecución de los objetivos del sprint. Esto permite que el equipo y el Scrum Master puedan tomar medidas correctivas rápidamente si es necesario.
7. Cultura de Retroalimentación: Fomentar una cultura donde la retroalimentación constructiva sea bienvenida es fundamental. Esto incluye no solo recibir retroalimentación sobre el trabajo realizado, sino también sobre cómo mejorar los procesos y la comunicación dentro del equipo

2.3 PROPUESTA DE ORGANIZACIÓN DEL TRABAJO.

Las epics serán un reflejo de los entregables del proyecto.
Los issues serán las historias de usuario que cada epic puede contener.
Las task van a hacer las veces de las tareas necesarias para completar una historia de usuario.
Los sprints tendrán una duración fija de 10 días en todo el proyecto, al inicio de cada srpint (día 1 del sprint) se celebrará la reunión de inicio de sprint y al final de cada día (día 10 del sprint) se llevarán a cabo las reuniones de review y retrospective.
Las daily meeting pueden flexibilizarse y utilizar alguna herramienta en línea para lleva el control de cada recurso del proyecto e ir plasmado en cada reunión daily:

- Trabajo de ayer.
- Trabajo que se realizara el día de ahora.
- Impedimentos para completar las tareas.

LA GERARQUIA PROPUESTA.
La jerarquía que seguirán lo elementos descritos anteriormente es la siguiente:

- EPICA
    - SRPINT
        - HISTORIA DE USUARIO

- TAREAS PARA TERMINAR LA HISTORIA DE USUARIO.

2.3.1.1 Complemento entre scrum y project.

Una épica en scrum es un hito en Project.
Una historia de usuario es una tarea padre en Project.
Una tarea de historia de usuario es una tarea hijo en Project.
Tanto un scrum board en azure boards como un documento Project, son compatibles.

2.3.1.2 Nomenclatura propuesta.

Para las épicas: Descripción breve del entregable completo según tabla 5.3 de entregables de los RFP, ejemplo: la épica 2 se titulará: Diseño de nuevo sistema

Para los Sprint: Sprint + índice secuencial + descripción breve del sprint, ejemplo: Sprint 1. Diagnóstico y plan de trabajo, para efectos demostrativos, se adjuntó un plan en Project únicamente con el nombre de Sprint: Sprint + índice secuencial

Para las historias de usuarios: colocar las iniciales de historia de usuario más un índice secuencial y la descripción de la historia de usuario, ejemplo: HU1. Como analista, Quiero realizar un diagnóstico del sistema actual, evaluar los riesgos, desarrollar una propuesta técnica y crear un plan de trabajo detallado, Para asegurar que las mejoras del sistema se implementen de manera organizada y efectiva.

Para las tareas: Colocar la descripción precisa de lo que la tarea pretende resolver, ejemplo: Reunión con DTIT MAG para evaluación del SISA y los módulos que se desean comunicar con MSF en SIECA
NOTA: Los ejemplos anteriores, son reales, tomados del plan de trabajo esbozado en esta propuesta de desarrollo.

2.4 CRONOLOGÍA Y RESULTADOS

2.4.1 Línea base y plan del proyecto.

El RFP nos proporciona una hoja de ruta clara en la definición de entregables del proyecto, como lo mencionamos anteriormente, hemos estructurado todo el proyecto para poder plasmar en MS Project las etapas de todo el desarrollo y lo complementaremos con la herramienta azure boards, a la hora de llevar el control cuando el desarrollo ya esté en marcha.

Con la información que ya tenemos de los RFP, podemos esbozar las primeras 2 épicas, sus sprints, historias de usuarios, tareas asociadas y sus respectivos tiempos.

Para las épicas, historias de usuario y tareas desde la épica 3 en adelante, si bien conocemos la épica, no conocemos con exactitud las historias de usuario asociadas, pero si conocemos los sprints asociados a la epica, dado que hemos definidos sprints de 10 días, dicho lo anterior tenemos el siguiente resumen.

2.4.2 Elementos identificados para el plan de trabajo.

2.4.2.1 Épicas / alcance.

1. Diagnóstico, evaluación de riesgos, propuesta técnica y plan de trabajo

2. Diseño de nuevo sistema
3. Ajustes al módulo normativo del SISA
4. Sistema de colas e interconexiones
5. Módulo de monitoreo del sistema de colas.
6. Soporte técnico para el uso del sistema.

Fecha de inicio del proyecto: 1 de julio de 2024.
Fecha de finalización del proyecto: 26 de mayo de 2025.

2.4.2.2 Historias de usuario y tareas asociadas identificadas.

- EPICA 1. HU1. Como analista, Quiero realizar un diagnóstico del sistema actual, evaluar los riesgos, desarrollar una propuesta técnica y crear un plan de trabajo detallado, Para asegurar que las mejoras del sistema se implementen de manera organizada y efectiva.
- EPICA 2. HU2. Como analista, quiero realizar el diseño de las pantallas que serán modificadas en el SISA en MAG.
- EPICA 2. HU3. Como analista, quiero realizar el diagrama de procesos.
- EPICA 2. HU4. Como analista, quiero mostrar en un diagrama y describir los cambios que sufrirá la base de datos.
- EPICA 2. HU5. Como analista, quiero realizar el diseño del sistema de colas
- EPICA 2. HU6. Como analista, quiero realizar el diseño del sistema de monitoreo de colas
- EPICA 2. HU7. Como analista, quiero describir los servicios web que expondrá la SIECA
- EPICA 2. HU8. Como analista, deseo describir el diseño de los clientes web que se comunicaran con los servicios web de la SIECA
- EPICA 2. HU9. Como analista, deseo construir el documento necesario para el cierre de la EPICA 2
- Desde la EPICA 3 a la EPICA 6. HISTORIAS DE USUARIO GENERICA: Se van a ver plasmadas en el plan del proyecto anexo de forma genérica, dado que necesitamos pasar por las épicas 1 y 2 para poder definir las historias de usuario subyacentes y sus correspondientes tareas.
    - HU. [rol], [funcionalidad], [beneficio].

Stakeholders identificados.

- ICT Business Analyst del Proyecto ATraCSI en El Salvador
- Técnico / Agente de Negocio MAG Asignado al proyecto.
- Técnico / Agente de la dirección de tecnología (DTIT) del MAG.

2.4.3 Plan de trabajo propuesto (ver enlace)

No.	Nombre del Sprint	Descripción	días	Épica asociada	días del entregable	RECURSO	Inicio	Fin	Observación
1	Srpint 1	Diagnóstico y plan de trabajo	10	1	10	**PM, DEV**	01-jul-24	12-jul-24	Inicio y **fin de épica 1 (ENTREGABLE 1)**
2	Srpint 2	Diseño del Sistema (Modificaciones)	10	2		**PM, DEV**	15-jul-24		Inicio de épica 2
3	Srpint 3	Servicios web, sistema de colas y monitoreo.	10	2	20	**PM, DEV**		09-ago-24	**Fin de épica 2 (ENTREGABLE 2)**
4	Srpint 4	A definir	10	3		**PM, DEV**	12-ago-24		Inicio de épica 3 y épica 6
5	Srpint 5	A definir	10	3		**PM, DEV**			
6	Srpint 6	A definir	10	3		**PM, DEV, QA**			
7	Srpint 7	A definir	10	3		**PM, DEV, QA**			
8	Srpint 8	A definir	10	3,6	50	**PM, DEV, QA**	07-oct-24	18-oct-24	Inicio épica 6, **fin de épica 3 ENTREGABLE 3**
9	Srpint 9	A definir	10	4, 5, 6		**PM, DEV, QA**	21-oct-24		Inicio de épica 4 y épica 5
10	Srpint 10	A definir	10	4, 5, 6		**PM, DEV, QA**			
11	Srpint 11	A definir	10	4, 5, 6		**PM, DEV, QA**			
12	Srpint 12	A definir	10	4, 5, 6		**PM, DEV, QA**			
13	Srpint 13	A definir	10	4, 5, 6		**PM, DEV, QA**			
14	Srpint 14	A definir	10	4, 5, 6		**PM, DEV, QA**			
15	Srpint 15	A definir	10	4, 5, 6		**PM, DEV, QA**			
16	Srpint 16	A definir	10	4, 5, 6		**PM, DEV, QA**			
17	Srpint 17	A definir	10	4, 5, 6		**PM, DEV, QA**			
18	Srpint 18	A definir	10	4, 5, 6		**PM, DEV, QA**			
19	Srpint 19	A definir	10	4, 5, 6	110	**PM, DEV, QA**		21-mar-24	**Fin de épica 4 ENTREGABLE 4**
20	Srpint 20	A definir	10	5, 6		**PM, DEV, QA**			
21	Srpint 21	A definir	10	5, 6		**PM, DEV, QA**			
22	Srpint 22	A definir	10	5, 6	140	**PM, DEV, QA**		02-may-25	**Fin de épica 5 ENTREGABLE 5**
23	Srpint 23	A definir	10	6		**PM, DEV, QA**			
24	Srpint 24	A definir	10	6	166	**PM, DEV, QA**		26-may-25	**Fin de épica 6 ENTREGABLE 6**
25	**Días totales del proyecto**				**236**				

LEYENDAS:

PM: Project Manager
DEV: Full Stack Developer
QA: Quality Assurance

NOTA: El Sprint 4, Épica 3 en adelante, depende de la información recabada en los sprints 1, 2 y 3, épicas 1 y 2 respectivamente.

2.5 PLANIFICACIÓN DE LA CALIDAD DEL PROYECTO.

La planificación de la calidad en el proyecto tiene como propósito medir la calidad de los productos desarrollados, en las fases de entrada, procesamiento y salida.
Los procesos de aseguramiento de calidad se aplicarán a los siguientes productos del proyecto:

- Módulos mejorados, y funcionalidades nuevas al sistema SISA para la comunicación con MSF de SIECA.
- Sistema de colas.
- Sistema de monitoreo de colas.

2.5.1 Asegurar la calidad del código.

Propuesta: Sonar QUBE.

SonarQube es una herramienta de revisión de código automática y autoadministrada que ayuda sistemáticamente a ofrecer código limpio. Como elemento central de una solución Sonar, SonarQube se integra en el flujo de trabajo existente y detecta problemas en el código para ayudar a realizar inspecciones continuas del código de los proyectos.

2.5.2 Objetivos del aseguramiento de la calidad del proyecto

1. Planificar las actividades de aseguramiento de la calidad de cada fase del proyecto.
2. Revisar y auditar objetivamente los productos y las actividades para verificar que están conformes con los procedimientos y estándares aplicables.
3. Proporcionar los resultados de estas revisiones o auditorías informando a la dirección cuando sea necesaria su mediación.

2.5.3 Metas del plan de calidad

- Planificar las actividades de SQA.
- Verificar la adherencia de los productos y actividades de software a los estándares, a los procedimientos y a los requisitos aplicables.
- Documentar y comunicar sobre las tareas que no cumplen con los estándares o procedimientos establecidos y que no se pueden resolver dentro del proyecto del software para que sean escaladas.

2.5.4 Estándares de calidad que se pueden tomar en consideración en el proyecto

- IEEE 730-1998 Planes de aseguramiento de la calidad del software.
- IEEE 829-1998 Documentación de pruebas del software.
- IEEE 982.1, 982.2 Diccionario estándar de medidas para producir software fiable.
- IEEE 1008-1987 Pruebas de unidad del software.
- IEEE 1012-1998 Verificación y validación del software.
- IEEE 1028-1997 Revisiones del software.
- IEEE 1044-1993 Clasificación estándar para anomalías del software.
- IEEE 1061-1992 Estándar para una metodología de métricas de calidad del software.
- IEEE 1228-1994 Planes de seguridad del software.

2.5.5 Métricas del desarrollo de los productos de software.

Las métricas van en lista de comprobación que se emplean para 'apuntar' atributos específicos del software. El esquema de puntuación es una escala del 0 (bajo) al 10 (alto) En donde se emplean las siguientes métricas en el esquema de puntuación:
Métrica de calidad.

- Corrección: Hasta dónde satisface un programa su especificación y consigue los objetivos de la misión del cliente.
- Fiabilidad: Hasta dónde puede quedarse un programa que lleve a cabo su función pretendida con la exactitud solicitada. Cabe hacer notar que se han propuesto otras definiciones de fiabilidad más completas.
- Eficiencia: El conjunto de recursos informáticos y de código necesarios para que un programa realice su función. - Integridad: Hasta dónde se puede controlar el acceso al software o a los datos por individuos no autorizados.
- Usabilidad (facilidad de manejo): El esfuerzo necesario para aprender, operar, y preparar datos de entrada e interpretar la salida (resultados) de un programa.
- Facilidad de mantenimiento: El esfuerzo necesario para localizar y arreglar un error en un programa. - Flexibilidad: El esfuerzo necesario para modificar un programa operativo.
- Facilidad de prueba: El esfuerzo necesario para aprobar un programa para asegurarse de que realiza su función pretendida.
- Portabilidad: El esfuerzo necesario para trasladar el programa de un entorno de sistema hardware y/o software a otro. - Reusabilidad: (capacidad de reutilización): Hasta dónde se puede volver a utilizar un programa (o partes) en otras aplicaciones con relación al empaquetamiento y alcance de las funciones que ejecuta el programa.
- Interoperabilidad: El esfuerzo necesario para acoplar un sistema con otro.
- Factor de calidad.
- Facilidad de auditoria: La facilidad con la que se puede justificar el cumplimiento de los estándares.
- Exactitud: La exactitud de los cálculos y del control.
- Estandarización de comunicaciones: El nivel de empleo de estándares de interfaces, protocolos y anchos de banda.
- Complexión: El grado con que se ha logrado la implementación total de una función.
- Concisión: Lo compacto que resulta ser el programa en términos de líneas de código.
- Consistencia: El uso de un diseño uniforme y de técnicas de documentación a través del proyecto de desarrollo del software.
- Estandarización de datos: El empleo de estructuras y tipos de datos estándares a lo largo del programa.
- Tolerancia al error: El deterioro causado cuando un programa descubre un error.
- Eficiencia de ejecución: El rendimiento del funcionamiento de un programa.
- Capacidad de expansión. El grado con que se pueden aumentar el diseño arquitectónico, de datos o procedimental.
- Generalidad: La extensión de aplicación potencial de los componentes de programa.
- Independencia del hardware: El grado con que se desacopla el software del hardware donde opera.
- Instrumentación: El grado con que el programa vigila su propio funcionamiento e identifica los errores que suceden.
- Modularidad: La independencia funcional de componentes de programa.
- Operatividad: La facilidad de operación de un programa.
- Trazabilidad: La capacidad de alcanzar una representación del diseño o un componente real del programa hasta los requisitos.

2.5.6 Tabla de métricas de calidad del software

La siguiente rubrica aplica para todos los módulos y funcionalidades nuevas y mejoras desarrollados en la consultoría.

Métrica de calidad del software / *Factor de calidad*	*corrección*	*fiabilidad*	*eficiencia*	*integridad*	*mantenimiento*	*flexibilidad*	*capacidad de pruebas*	*portabilidad*	*reusabilidad*	*interoperabilidad*
facilidad de auditar										
exactitud										
estandarización de comunicaciones										
compleción										
complejidad										
concisión										
consistencia										
estandarización de datos										
tolerancia a errores										
Eficiencia de ejecución										
capacidad de expansión										
generalidad										
interdependencia del hardware										
instrumentación										
modularidad										
operatividad										
seguridad										
auto documentación										
simplicidad										
independencia del sistema										
trazabilidad										
facilidad de información										

3 RENDIMIENTO PASADO DEL OFERENTE.

AIT Company inicio sus operaciones en febrero de 2024, con la idea de generar soluciones informáticas a problemas cotidianos a los que se enfrentan las Instituciones del Estado y a propietarios de negocios ya sean pequeños, medianos o grandes.

Los fundadores y el personal de AIT tienen una basta experiencia en proyectos con Instituciones del Estado que pueden verse reflejadas en las competencias técnicas en el siguiente apartado (experiencia y capacidad técnica). Por su parte como AIT, se han desarrollado las siguientes iniciativas:

1. Proyecto de Sistema de Facturación Electrónica: El proyecto consiste en la creación de un aplicativo web diseñado para llevar el control de inventarios y facturación de micro, pequeña y medianas empresas y que posteriormente son enviadas de forma automática al Ministerio de Hacienda; por tanto, el aplicativo se conecta al MdH, se obtiene sello de recepción y por último enviar las facturas a los consumidores finales (clientes de los comercios).

2. Proyectos de desarrollo de facturación electrónica: Actualmente se esta trabajando con desarrollos a la medida que permiten la interconexión de los sistemas de facturación de cualquier comercio, con los sistemas del ministerio de hacienda, para que permitan generar la factura electrónica de forma automática.

4 EXPERIENCIA Y CAPACIDAD TECNICA.

4.1 Project Manager.

Candidato propuesto: Franklim Alexander Arevalo.

- Ingeniero Industrial.
- Máster en finanzas.
- Project Manager certificado por Project Management Institute (PMI).

Experiencia en dirección de proyectos.

Banco de América Central:

(2023 - 2024), Vicepresidencia de Finanzas y Operaciones.

PROYECTO: Digitalización del 100% de documento de la compañía en operaciones contables generando ahorros en más de US$ 30,000 anuales.

Posición desempeñada: Project Manager.

Descripción del proyecto: La compañía dentro de su visión de digitalización requería eliminar todos los flujos físicos que a su vez representaban más de ¾ de millón de impresiones al año, mas el resguardo físico de esa de esos papeles por un periodo de 10 años (gastos por alquiler de almacenamiento). El proyecto consistió en:

- ✓ Sustituir el flujo de trabajo físico por un flujo digital por medio de una herramienta informática que utilizan más de 400 usuarios.
- ✓ Eliminar todas las firmas manuscritas, sellos estampados, y codificaciones manuscritas realizadas en los flujos de trabajo físicos.
- ✓ Aumentar la eficiencia operativa eliminando horas de trabajo (con la digitalización) y disminuyendo los riesgos inherentes al control de documentos físicos.
- ✓ Adquisición de un repositorio documental para el resguardo seguro de la información.

Banco Central de Reserva:

(2015-2019), Centro de Tramites de Importaciones y Exportaciones

PROYECTO: Proyectos de Facilitación del Comercio Exterior.

Descripción de los proyecto: Participe en la formulación, planificación, la gestión de recursos y la dirección de proyectos relacionados con las interconexiones del Sistema de Facilitación de Comercio Exterior (SFCE) con los sistemas del Ministerio de Hacienda, Ministerio de Salud, Ministerio de Agricultura y Ganadería, Ministerio de la Defensa Nacio nal, Ministerio de Medio Ambiente y la Dirección Nacional de Medicamentos; lo anterior con el objetivo de centralizar, agilizar y simplificar las gestiones legalmente establecidas para el registro, autorización y emisión de los documentos de las operaciones de importación y exportación.

4.2 Desarrollador Full Stack Senior.

Candidato propuesto: Jonathan Arevalo Guevara.

- Ingeniero en ciencias de la computación.
- Scrum Master Certificado.

Experiencia en puesto similares:

Trabajando desde 2018 con diferentes organismos internacionales.

Cooperación FOMILENIO II.

(2018 - 2021), Ministerio de Agricultura y Ganadería de El Salvador, otros.

PROYECTO: Implementación de mejoras y sistematización de los procesos de comercio exterior de El Salvador: Contratación de desarrolladores informáticos para construir e implementar sistemas de información orientados a la facilitación de comercio.

Posición desempeñada: (Consultoría) Arquitecto de software / Líder técnico / Senior Developer.

Cabe destacar que Jonathan Arevalo Guevara se encuentra desempeñando un rol de apoyo desde el 2018 a la fecha, al ministerio de Agricultura y ganadería, en diferentes programas de cooperación, fue parte de la primera migración de tecnologías desde los módulos del SISA legado al nuevo SISA con tecnologías de vanguardia en 2018, estuvo en el levantamiento del nuevo registro único agropecuario y la nueva plataforma de servicios en línea del MAG.

Las instituciones a las que envolvió la cooperación Fomilenio II fueron:

(** tuvo participación en el desarrollo de funcionalidades)

- **Banco Central de Reserva de El Salvador (Senior developer) **.**
- **Ministerio de Agricultura y Ganadería (Líder técnico / Arquitecto de software) **.**
- **Ministerio de Hacienda (Senior Developer) **.**
- **Ministerio de Medio Ambiente y Recursos Naturales (Senior Developer) **.**
- Ministerio de Defensa.
- Centro Nacional de Medicamentos.
- **Ministerio de Salud (Senior Developer) **.**
- **Organismo Internacional Regional de Sanidad Agropecuaria (Senior Developer) **.**
- Centro Nacional de Registros.

The U.S. Department of the Interior's International Technical Assistance Program (DOI-ITAP).

(2022 – 2023), Ministerio de Agricultura y Ganadería de El Salvador.

Sistema de gestión de comercio de vida silvestre y certificado CITES.

Posición desempeñada: Consulto de software, Full Stack Developer.

Ministerio de agricultura y ganadería de El Salvador:

(2022).

Automatización de proceso de firmas para entrega de paquetes alimenticios

Posición desempeñada: Consulto de software, Full Stack Developer.

NATHAN INC. / USAID

(2021 – julio 2024).

(2021 – 2023), Misterio de Salud de Guatemala, Líder técnico / Senior Software Developer

(2024), Ministerio de Agricultura y Ganadería, El Salvador, Senior Software Developer

Resumen de experiencia en el MAG.

Los módulos del Ministerio de Agricultura y ganadería en los que Jonathan ha tenido partición en desarrollos anteriores:

- (2018 - 2021) SISA (SISTEMA DE INFORMACION DE SANIDAD AGROPECUARIA).
- (2018 - 2021) SERVICES (PLATAFORMA DE SERVICIOS EN LINEA).
- (2018 - 2021) RUA (REGISTRO UNICO AGROPECUARIO).
- (2022 - 2023) CITES (SISTEMA DE COMERCIO EXTERIOR Y VIDA SILVESTRE).
- (2023) SISTEMA DE SUBSUDIO AGROPECUARIO (ENTREGA DE PAQUETES).
- (2023 – 2024) SISTEMA DE PRECERTIFICADOS.
- (2043 – 2024) FRONTERA.
- (2024) Migración y actualización de productos Fito y zoo sanitarios de enmienda 5 a enmienda 7 para MSF SIECA en su primera etapa.

4.3 Ingeniero de Quality Assurance (QA) Senior.

Candidato propuesto: Harold Steev Gómez Martínez.

- Licenciado en ciencias de la computación.

Experiencia en puesto similares:

USAID Regional Trade Facilitation and Border Management Project Implemented by NATHAN Associates Inc.

(2020 – agosto 2024), Ministerio de Agricultura y Ganadería, El Salvador.

Jefe de QA para la región para el proyecto de facilitación de comercio exterior (El Salvador, Honduras y Guatemala.

- Coordinar las actividades de pruebas para los proyectos de la región con la

- Implementación de metodología de pruebas en base a estándares.
- Realizar cálculos de tiempos para generar los planes de trabajo los diferentes equipos de la región.
- Coordinar con las diferentes contrapartes de los proyectos a implementar
- Capacitaciones de metodología de pruebas, generación de casos de pruebas, se les
- Proporciona sugerencias que deberían implementar para tener control sobre pruebas
- Realizar pruebas automatizadas una vez el aplicativo se encuentre estable
- Coordinar actividades de pruebas piloto con todas las contrapartes para llevar el proyecto a aceptaciones y aprobaciones de usuario.
- Revisar informas de avance de los diferentes proyectos
- Presentar métricas y KPI por cada proyecto de la región para informar que se esta finalizado en base a casos de uso o historias de usuario.
- Participar en reuniones de definición de los diferentes proyectos con la contraparte para facilitar la implementación de artefactos de pruebas para los diferentes equipos

Banco central de reserva

(2019 al 2020)

Analista de pruebas QA.

- Continuidad de proyecto de Fomilenio II para integraciones de aplicativos con diferente entes de gobierno aplicativo de importaciones
- Realizar plantillas de requerimientos para modificar los aplicativos y realizar integraciones con diferentes instituciones de gobierno
- Identificación de los posibles riegos del proyecto
- Realizar pruebas Automatizadas mediante script con selenimun y Katalon los ide y pruebas manuales a los aplicativos de Banco central de reserva y creación de la nueva plataforma
- Asesoría al banco para implementar metodologías de pruebas con Eclipse y selenimun
- Realizar instalaciones de Testlink para documentación de pruebas.
- Documentar fallos y continuar con la implementación de la metodológica de pruebas para el proyecto
- Aplicación de Scrum en los diferentes desarrollos

Fomilenio II

(2019 al 2020), **Banco central de reserva**

Quality Assurance Senior.

Coordinar equipo de QA

- Identificación de los posibles riegos del proyecto
- Realizar pruebas Automatizadas mediante script con selenimun y Katalon los ide y pruebas manuales a los aplicativos de Banco central de reserva y creación de la nueva plataforma
- Asesoría al banco para implementar metodologías de pruebas con Eclipse y selenimun
- Realizar instalaciones de Testlink para documentación de pruebas.
- Documentar fallos e implementar la metodológica de pruebas para el proyecto
- Aplicación de Scrum en los diferentes desarrollos

5 ANEXOS.

5.1 Plan del proyecto en project.

- Plan.mpp, necesita Microsoft Project para abrirlo.

5.2 Full Stack Developer:

- Hoja de vida. Ver adjunto
- Atestados. Ver adjunto
- Constancias y contratos:
    - Fomilenio, Ver adjunto
    - Intecjuba DOI-ITAP
        - Contrato. Ver adjunto
        - Adenda. Ver adjunto
    - USAID, Nathan Inc.
        - Contrato 1. Ver adjunto
        - Adenda 1. Ver adjunto
        - Contrato 2. Ver adjunto
        - Adenda Contrato 2. Ver adjunto
        - Contrato 3. Ver adjunto

5.3 Quality Assurance Senior.

- Hoja de vida. Ver adjunto
- Documentos y referencias. Ver adjunto
- Referencia Fomilenio. Ver adjunto
- Diploma SCRUM. Ver adjunto
- Titulo. Ver adjunto

5.4 Project Manager.

- Hoja de vida, Ver adjunto
- Atestados, Ver adjunto
- Certificación PMI, Ver adjunto

NOTA: puede descargar todos los archivos de la propuesta técnica, en el siguiente enlace:
Descargar archivos adjuntos.