UNIVERSIDAD TÉCNICA DEL NORTE

Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Ingeniería en Sistemas Computacionales

ELABORACIÓN DE UNA GUÍA METODOLÓGICA PARA LA GESTIÓN DE PROYECTOS DE SOFTWARE UTILIZANDO LA HERRAMIENTA GITLAB Y LA METODOLOGÍA SCRUM PARA FORTALECER LOS PROYECTOS DE DESARROLLO DE SOFTWARE DE LOS ESTUDIANTES DE LA CARRERA DE INGENIERÍA DE SOFTWARE.

Trabajo de Grado previo a la obtención del título de Ingeniero en Sistemas Computacionales

Autor:

Erick Alexander Vásconez Endara

Director:

Ing. MSc. Mauricio Xavier Rea Peñafiel

Ibarra – Ecuador

2022

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1004159784		
APELLIDOS Y NOMBRES:	VÁSCONEZ ENDARA ERICK ALEXANDER		
DIRECCIÓN:	Caranqui, Huiracocha 7-118 y Nazacota Puento		
EMAIL:	eavasconeze@utn.edu.ec		
TELÉFONO FIJO:	-	**TELÉFONO MÓVIL:**	0985729425

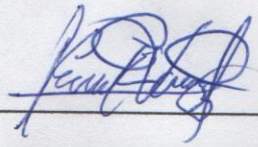
DATOS DE LA OBRA	
TÍTULO:	**ELABORACIÓN DE UNA GUÍA METODOLÓGICA PARA LA GESTIÓN DE PROYECTOS DE SOFTWARE UTILIZANDO LA HERRAMIENTA GITLAB Y LA METODOLOGÍA SCRUM PARA FORTALECER LOS PROYECTOS DE DESARROLLO DE SOFTWARE DE LOS ESTUDIANTES DE LA CARRERA DE INGENIERÍA DE SOFTWARE.**
AUTOR (ES):	Erick Alexander Vásconez Endara
FECHA: DD/MM/AAAA	05/05/2022
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	[x] **PREGRADO** [ ] **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniero en Sistemas Computacionales
ASESOR /DIRECTOR:	Msc. Mauricio Rea

1. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 6 días del mes de mayo de 2022

EL AUTOR:



Erick Alexander Vásconez Endara
1004159784

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN DEL DIRECTOR

Por medio de la presente yo Msc. Mauricio Rea, certifico que el Sr. Erick Alexander Vásconez Endara, portador de la cédula de identidad Nro.100415978-4, ha trabajado en el desarrollo del proyecto de tesis "**ELABORACIÓN DE UNA GUÍA METODOLÓGICA PARA LA GESTIÓN DE PROYECTOS DE SOFTWARE UTILIZANDO LA HERRAMIENTA GITLAB Y LA METODOLOGÍA SCRUM PARA FORTALECER LOS PROYECTOS DE DESARROLLO DE SOFTWARE DE LOS ESTUDIANTES DE LA CARRERA DE INGENIERÍA DE SOFTWARE.**", previo a la obtención del título de Ingeniero en Sistemas Computacionales, lo cual ha realizado en su totalidad con responsabilidad.

Es todo cuanto puedo certificar en honor de la verdad.

Atentamente:

firmado electrónicamente por:
XAVIER MAURICIO REA PEÑAFIEL

Msc. Mauricio Rea

DIRECTOR DE TESIS

Dedicatoria

Este trabajo se lo dedico a mi familia y en especial a mi madre Marcia por todo el trabajo, sacrificio y su amor incondicional que me ha brindado en cada etapa de mi vida y sobre todo por formarme como persona y como profesional.

Erick Vásconez

Agradecimientos

Agradezco a mi familia en especial a mi madre por ser pieza fundamental en el logro de este objetivo y estar siempre a mi lado en todo momento.

A mis amigos que siempre han estado dispuestos a colaborar en cualquier problema que se ha presentado.

A mi tutor, por estar pendiente desde el inicio hasta el final del proceso de este trabajo y brindarme todo su apoyo.

Erick Vásconez

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Resumen

En la carrera de Ingeniería de Software de la Universidad Técnica del Norte, las malas prácticas dentro de los proyectos de aula desarrollo de software recaen en problemas especialmente en la fase de desarrollo y posteriormente en la implementación del software. Esto se evidencia porque no se realiza una gestión de proyectos durante todo le proceso de desarrollo de software. Los estudiantes no logran cumplir los principios de las metodologías y no acompañan a esto herramientas que automaticen el proceso de gestión de proyectos de software. Al no implementar en conjunto una metodología con una herramienta automatizada hace que el desarrollo de software incumpla la planificación indicada, aumento de recursos de costo y tiempo y por ende el incumplimiento y/o fracaso del proyecto.

El presente trabajo de titulación enfoca sus esfuerzos en describir los pasos para gestionar el desarrollo de software por medio de la elaboración de una guía metodológica que contendrá un marco teórico y tecnológico sobre técnicas de uso en la gestión de proyectos orientada a la metodología Scrum con la herramienta GitLab. La guía metodológica contiene una estructura conformada por una introducción, objetivos, alcance, terminología, responsables, descripción de las actividades y formatos- diagramas de flujo.

Para la validación de la guía metodológica se realizó una prueba de concepto sobre una aplicación web para validación de la norma ISO 27012. La prueba de concepto fue desarrollada aplicando la guía metodológica, la misma que dentro de sus actividades define tres fases que son: Pre-Juego, Juego o Desarrollo y Post-Juego; y se aplicó la escala de Likert para obtener un porcentaje del éxito de cumplimento, en el cual se obtuvo un 88.8 % de éxito.

Finalmente se aplicó una encuesta SUS para validar la usabilidad de la prueba de concepto de lo cual se obtuvieron respuestas positivas con respecto al sistema.

Palabras clave: Gestión de proyectos, scrum, gitlab, guía, metodología

Abstract

In the Software Engineering career at the Universidad Técnica del Norte, bad practices within software development classroom projects fall into problems, especially in the development phase and later in the implementation of the software. This is evidenced because project management is not carried out throughout the software development process. The students fail to comply with the principles of the methodologies and do not accompany this with tools that automate the software project management process. By not implementing a methodology together with an automated tool, the software development fails to comply with the indicated planning, increases cost and time resources and therefore the non-compliance and/or failure of the project.

This degree work focuses its efforts on describing the steps to manage software development through the development of a methodological guide that will contain a theoretical and technological framework on techniques of use in project management oriented to the Scrum methodology with the GitLab tool. The methodological guide contains a structure made up of an introduction, objectives, scope, terminology, people in charge, description of the activities and formats-flow diagrams

For the validation of the methodological guide, a proof of concept was carried out on a web application for validation of the ISO 27012 standard. The proof of concept was developed by applying the methodological guide, the same that within its activities defines three phases that are: Pre-Game, Game, Post-Game; and the Likert scale was applied to obtain a percentage of compliance success, in which an 88.8% success rate was obtained.

Finally, a SUS survey was applied to evaluate the usability of the proof of concept, from which positive responses were obtained regarding the system.

Keywords: Project management, scrum, gitlab, guide, methodology

INTRODUCCIÓN

Antecedentes

En el pasado, muchos proyectos de software fallaron antes de que se completara la etapa, se debe a la falta de planificación y falta de proyecto técnicas y herramientas de gestión. Una buena gestión de proyectos siempre conducirá al éxito del software (Hayat et al., 2019)

El Standish Group en los reportes publicados en los últimos 5 años muestra que el comportamiento de los proyectos exitosos sigue siendo menor que los proyectos cancelados y fallidos. Una de las causas es la: Identificación de defectos en las etapas de pruebas (Diaz et al., 2019)

En la carrera de ingeniería de software de la Universidad Técnica del Norte, dentro de los proyectos de aula de los estudiantes, no se consolida un seguimiento integral a las fases del ciclo de vida del software, es decir, muchos proyectos no cumplen las especificaciones requeridas, no se culminan los proyectos en el período establecido , no se sigue una planificación desde la concepción del proyecto hasta la etapa final de implementación, esto se evidencia por diferentes razones principales como: la detección tardía de errores en las fases finales, ausencia de pruebas funcionales, no trabajar en equipo de manera sincrónica en la codificación, etc.

Esta incidencia también se ve reflejada en la elaboración de las tesis dedicadas al desarrollo de software, en las cuales los tesistas no aplican herramientas automatizadas que permiten detectar a tiempo estos factores de riesgo y que permiten tomar el control de gestión en todo el proceso de ciclo de vida del software, en consecuencia, no se cumple la planificación indicada y esto ocasiona el aumento recursos como costos y tiempo. No obstante, más allá del desconocimiento o no aplicabilidad del entorno tecnológico para la automatización de gestión de los proyectos de software, los estudiantes y tesistas no logran comprender la teoría y los principios de las metodologías y en mucho de los casos terminan aplicado de forma errónea y por ende el fracaso del proyecto.

Situación Actual

La carrera de ingeniería de software está conformada por 8 niveles de carrera durante los cuales se desarrollan proyectos de software en las diferentes asignaturas correspondientes y a su vez, estos en el transcurso de cada nivel van aumentando la complejidad y necesitan cumplir con todo el ciclo de vida del software. Muchos de los

estudiantes no integran herramientas que automaticen el desarrollo de software de sus proyectos, los equipos de trabajo unen sus módulos de código de forma manual porque no integran un control de versiones que sincronice sus fragmentos de código, esto provoca malestar en tiempos y en conflictos que aparecen en la etapa de pruebas.

Los estudiantes programan sin documentar la planificación y no siguen los requisitos, es decir que primero desarrollan y después documentan.

A pesar de los avances en las metodologías de desarrollo, la seguridad, siempre es una asignatura pendiente, en los flujos de trabajo tradicionales, la seguridad se aplica al final del SDLC (Ciclo de Vida del Desarrollo de Software) o al encontrarse el software en entornos productivos provocando un aumento significativo de los plazos de entrega y un aumento considerable en el coste de corrección de las vulnerabilidades encontradas, debido a la complejidad de implementar una solución en un entorno ya desplegado tanto en la búsqueda de la vulnerabilidad como su corrección (Iriz Ricote, 2019)

Planteamiento del Problema

En la carrera de Ingeniería de Software de la Universidad Técnica del Norte, las malas prácticas dentro de los proyectos de aula y la elaboración tesis de grado en cuanto al desarrollo de software recaen en un alto índice de errores en las fases de construcción, pruebas e implementación del software, no se realiza una gestión aplicada a todo el ciclo de vida del software. Los estudiantes no logran cumplir los principios de las metodologías y no acompañan a esto herramientas que automaticen el proceso de gestión de proyectos de software.

Al no implementar en conjunto una metodología con una herramienta automatizada hace que el desarrollo de software incumpla la planificación indicada, aumento de recursos de costo y tiempo y por ende el incumplimiento y/o fracaso del proyecto.

Por otro lado, las metodologías ágiles, como es el caso de la metodología Scrum, desarrollada en los años 90’s por Jeff Sutherland y Ken Schwaber, no manejan una documentación suficiente, que conduzca el proceso a través del ciclo de vida del desarrollo del software al igual que no disponen de artefactos tales como Diagramas y/o Casos de Uso, que evidencien el progreso en el proceso de desarrollo (Renato, 2016)

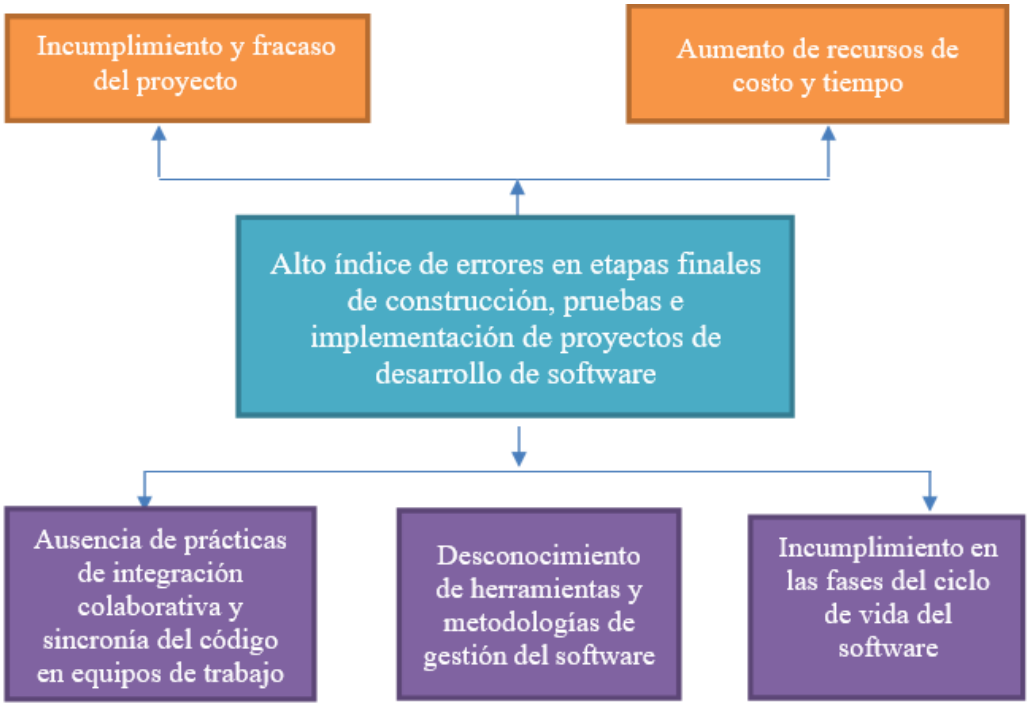


Figura 1: Árbol de problemas
Fuente: Propia

Objetivos

Objetivo General

Elaborar una guía metodológica para la gestión de proyectos de software utilizando la herramienta GitLab y la metodología Scrum para fortalecer los proyectos de desarrollo de software de los estudiantes de la carrera de Ingeniería de Software.

Objetivos Específicos

- Establecer un marco teórico sobre diferentes técnicas para la elaboración de guías metodológicas.
- Estudiar la herramienta GitLab y marco de trabajo Scrum para elaborar la guía metodológica.
- Validación de la guía mediante una prueba de concepto

Alcance

En la presente propuesta de tesis se elaborará una guía metodológica que contendrá un marco teórico y tecnológico sobre técnicas de uso en la gestión de proyectos de desarrollo de software orientada a la metodología Scrum y con la herramienta GitLab que permitirá gestionar la entrega ágil de software durante el proceso del ciclo de vida del software, es

decir: requerimientos, diseño, desarrollo o construcción, pruebas, despliegue o implementación y mantenimiento u operaciones.

Se desarrollará un sistema como prueba de concepto que permitirá a los usuarios acceder y seleccionar un checklist de características de la norma ISO 25012 sobre la calidad de producto de datos. Para la validación del software se realizará una encuesta de escala de usabilidad del sistema SUS (System Usability Scale).

La escala de usabilidad del sistema (SUS) proporciona una herramienta fiable y "rápida y sucia" para medir la usabilidad. Consiste en un cuestionario de 10 puntos con cinco opciones de respuesta para los encuestados; firmemente de acuerdo a fuertemente en desacuerdo (Usability.gov, 2013)

Se realizará la validación de la guía mediante un sistema (prueba de concepto) que utilizará las tecnologías siguientes:

- IDE Visual Studio Code
- Backend: Nodejs Framework Express
- Frontend: Vuejs
- Base de datos: MongoDB

El sistema contará con los siguientes módulos:

- Módulo de acceso usuarios
- Módulo listado de características ISO 25012 Las funcionalidades que tendrá el sistema son:Registro e ingreso de usuarios
- Seleccionar y guardar opciones de checklist ISO 25012.

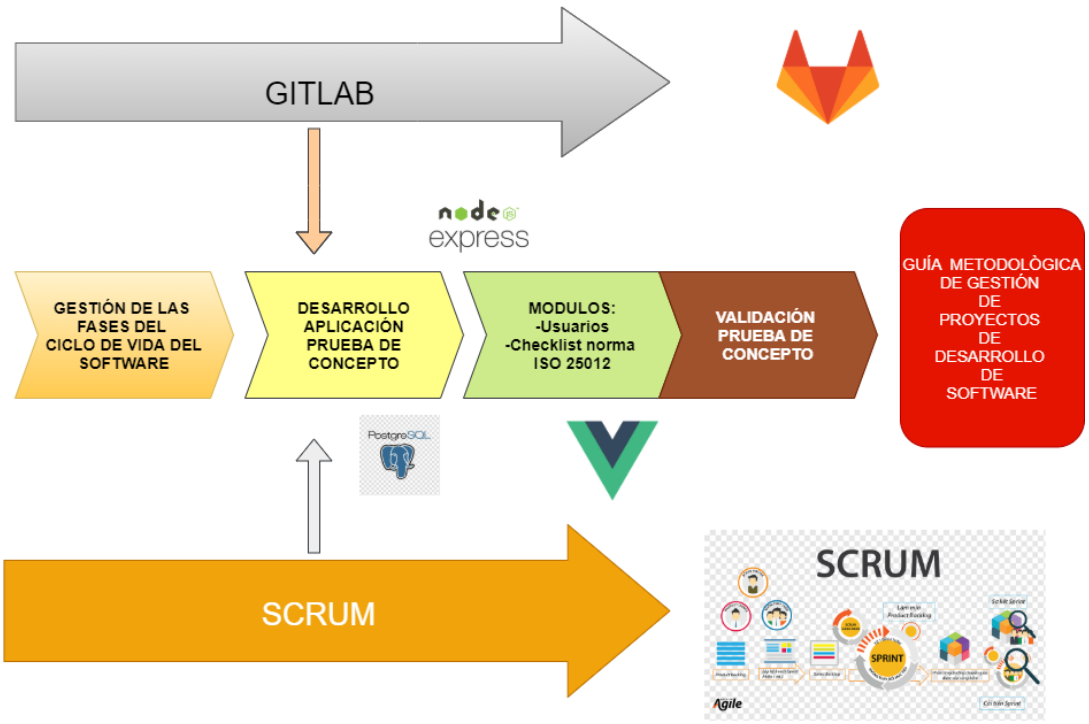


Figura 2 Alcance
Fuente: Propia

Existen múltiples herramientas y plataformas enfocadas a la mejora de gestión de proyectos de software.

La metodología ágil se adapta a los cambios de requisitos en el futuro, y tiene tal flexibilidad que puede manejar el costo, el alcance, calidad del software según las necesidades del cliente. Scrum es el marco de metodología ágil ya que se enfoca en el día a día gestión de proyectos y es más ampliamente adoptar el método de gestión de proyectos ágil (Hayat et al., 2019)

GitLab es una aplicación para el ciclo de vida del desarrollo de software. Está diseñado para gestionar la totalidad de un proyecto de software, incluida la gestión del código fuente, la gestión de proyectos, el control de calidad y la gestión de versiones. GitLab y su cadena de herramientas pueden soportar todo el ciclo de “DevOps” (Lehtovirta, 2017)

GitLab contiene muchas herramientas para crear, construir, monitor y administrar un proyecto. Y esto se hace de una manera en la que el equipo de operaciones no se ve obligado a utilizar las herramientas integradas (Mikko Dittmer, 2019)

Metodología

El método de investigación que se utilizará será la investigación documental para conocer la problemática académica, en la cual se realizará una revisión de la literatura, se utilizará la encuesta como técnica de investigación y finalmente definir un marco teórico y tecnológico sobre la plataforma GitLab y la metodología Scrum.

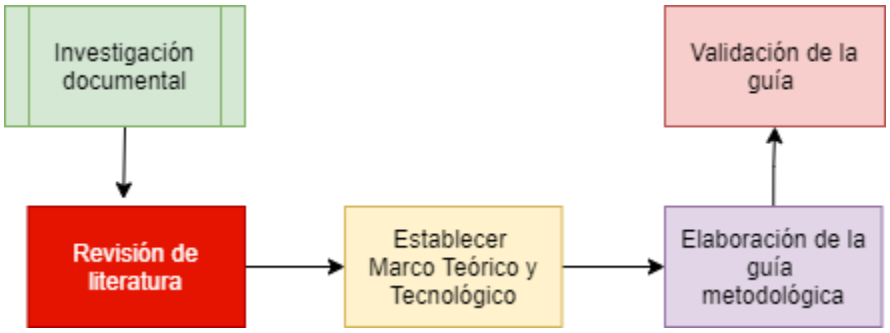


Figura 3: Proceso metodológico
Fuente: Propia

Justificación

Hoy en día la gestión de proyectos se está generalizando cada vez más en las actividades de las organizaciones; sin embargo, muchas compañías, ya sean grandes o pequeñas, tienen dificultades frente a cómo deben gestionar sus proyectos. La ausencia de buenas métricas para gestionar todo el ciclo del proyecto impide avances, optimización de tiempo, recursos, veracidad y conformidad que el cliente adquiere frente a lo que se le está ofreciendo, entre otros aspectos decisivos para el beneficio de la empresa (Patricia & Guerrero, 2016).

La importancia de esta investigación permite apoyar a las buenas prácticas en el proceso de desarrollo de software (gestión de proyectos) y dotar de un marco teórico y tecnológico, en que, de forma clara, ordenada y concisa se desempeñen los roles y las funciones asignadas al personal involucrado, a su vez de que sea un instrumento de cumplimiento y validez para lograr optimizar recursos de tiempo y costo durante la planificación hasta la implementación del proyecto.

El uso de una guía de gestión de proyectos brinda un marco de lineamientos específicos enfocados a los proyectos de aula de desarrollo de software de los estudiantes y para la elaboración de tesis, de esta manera se pretende contextualizar un ambiente metodológico que fomente estrategias y bases para el proceso, teniendo como beneficiarios

directos a la comunidad estudiantil, tesistas y docentes de la carrera de ingeniería de software y a fines.

En consecuencia, la presente propuesta pone a consideración la puesta en marcha la contemplación de los objetivos ODS:

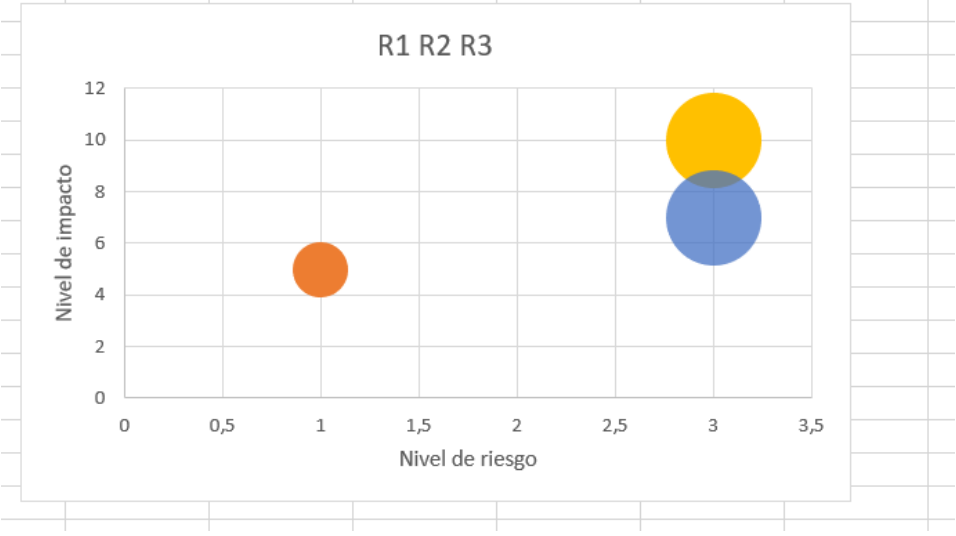
Objetivo 4: Educación de calidad: La entrega del trabajo propuesto servirá de instrumento para la contribución de conocimiento con estrategias nuevas a implementar en los proyectos de software.

Objetivo 9: Industria, innovación e infraestructura: Dotar de apoyo con nuevo conocimiento que contribuya al desarrollo e investigación tecnológica.

El uso de un marco de trabajo o guía en el proceso de desarrollo de software ayuda a que el personal involucrado tenga claridad en las actividades y roles desempeñados en el proceso, así como también que herramientas y recursos disponen para el cumplimiento efectivo y ordenado de sus actividades, reduciendo costo y tiempo de implementación (Guerrero & Guevara, 2019).

Riesgos

Riesgo	Impacto	Probabilidad
R1	10	3
R2	5	1
R3	7	3



RIESGO	Descripción del riesgo	Mitigación del riesgo
R1	Costos altos en nuevas características de GitLab	Verificar características prioritarias a utilizar
R2	Desconocimiento de nuevas funcionalidades de GitLab	Optar por un curso online, sea gratuito o pagado.
R3	Adquirir el estándar ISO solicitado para uso del sistema.	Buscar recursos en internet que contengan las métricas de la norma

Figura 4: Matríz de riesgos

Fuente: Propia

CAPÍTULO I

Marco Teórico

1.1. Fundamentos para elaborar una guía metodológica

Para la recopilación de información de la presente investigación se ha utilizado una búsqueda o mapeo sistemático de literatura (SMS).

El objetivo de esta revisión es mostrar una perspectiva general del campo científico, áreas de interés, focos y tendencias de los investigadores de la ingeniería de software. Es una revisión menos exhaustiva que una revisión sistemática de literatura cuyo fin es generar cuerpos de conocimiento. (Carrizo et al., 2018, párr.16)

A continuación, se describen los pasos para el proceso de revisión de mapeo sistemático según (Carrizo et al., 2018, párr.17):

Este proceso sistemático se basa en un modelo que define tres partes fundamentales en la investigación:

- Definición para la búsqueda: pregunta de investigación, alcance de la revisión, los criterios de inclusión y exclusión y por último la cadena de búsqueda.
- Ejecución de la búsqueda: definición de trabajos primarios y difusión de criterios de análisis.
- Discusión de resultados: se define os esquemas de caracterización y se analizan os resultados.

Los motores de búsqueda, repositorios bibliográficos analizados y las cadenas de búsqueda utilizadas se muestran en la Tabla 1.

Tabla 1: Repositorios y bases de datos bibliográficas y cadenas de búsqueda utilizadas

Fuente bibliográfica	Cadena de búsqueda
IEEE Xplorer	(“All Metadata”:methodological guideline) OR (“All Metadata”:guide)
Microsoft Academic	“(methods OR methodological) AND (“guides”)”
Google Scholar	“(methods OR methodological) AND (“guideline” OR “guide” OR “handbook”)” “(guías OR técnicas) AND (“metodologicas”)” ;

	“(“Cómo”) AND (”Crear“) AND (“Manual”) AND (“software OR “Procedimientos” OR “Usuario”)”
Scopus	“(methods OR methodological) AND (“guidelines” OR “guides” OR “handbook”)”

Fuente: Propia

A continuación, se presenta el resultado de la revisión de literatura.

1.1.1 Elementos de una guía o manual

De acuerdo con el resultado de la investigación se presenta la estructura y elementos para la elaboración de una guía metodológica según los diferentes autores mencionados en la Tabla 2.

Tabla 2: Elementos de una guía o manual

Autor	Tipo de documento	Partes, estructura, elementos
Según los autores **(Ordoñez Ortiz & Valle Caicedo, 2014)**, en su guía titulada “Guía del Emprendedor para negocios de TI”, propone un formato de pasos a seguir.	Guía	• Objetivo • Descripción • Artefactos- técnicas • Entregables
De acuerdo con los autores (Ochoa & Herrera, 2018), en su propuesta de un manual de políticas	Manual	• Introducción • Objetivos • Alcance • Definiciones principales • Responsabilidad • Desarrollo
Según el autor (Quezada, 2013) establece un manual de procedimientos.	Manual	• Título • Código • Introducción • Organización • Procedimientos • Responsabilidad • Medidas de Seguridad
Según el autor (Guzmán, 2018) afirma que todo manual de procedimientos debe estar definido de acuerdo según las	Manual	• Introducción • Objetivos • Alcance • Terminología • Responsables

necesidades de la entidad en cuestión.		• Descripción de las actividades • Formatos – Diagramas de flujo
Según los autores (Fincowsky & Benjamín, 2009) el manual debe tener la siguiente estructura.	Manual	• Identificación • Introducción • Índice • Contenido • Alcance de los procedimientos • Responsables • Normas de operación • Procedimiento (Descripción de las operaciones) • Diagramas de flujo • Formularios o formatos • Glosario de términos

Fuente: Propia

De acuerdo con la investigación realizada, se ha optado por la estructura del manual del autor Guzmán porque contiene los Formatos y diagramas de flujo, los mismos que serán utilizados en el desarrollo de la guía.

1.1.2. Propuesta metodológica

De acuerdo con el autor(Guzman, 2016) en su propuesta metodológica usando Scrum y PMBOK presenta una estructura de desarrollo de la Metodología:

- Descripción de la Fase del proyecto
- Gráfica del proceso de la fase del proyecto
- Propósito, Herramientas, Técnicas y roles que participan en cada proceso y subproceso que conforman la fase del proyecto
- Procedimiento por cada proceso y subproceso, donde se listan las actividades, tareas e incluye los roles responsables, entradas y salidas.
- Finalmente, la gráfica de cada proceso y subproceso
- Entregables

1.2. Ciclo de vida del software

La norma ISO 12207 define el Ciclo de Vida del Software como un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización de su uso. Existen distintos modelos de Ciclo de Vida, que determinan cuales son y de qué manera se ejecutan las etapas del desarrollo de software. Cada modelo presenta sus propias características y alcances de estas etapas. Para este artículo hemos considerado las siguientes etapas: Expresión de Necesidades y Especificaciones (Requerimientos), Análisis, Diseño, Implementación, Pruebas y Mantenimiento (Diéguez & Cares, 2011).

1.2.1. Etapas del ciclo de vida

Según (Iriz Ricote, 2019), las etapas del ciclo de vida son las siguientes:

- Planificación. Análisis y requisitos: El SDLC comienza con la fase de análisis de requisitos, donde los interesados discuten los requisitos del software que debe desarrollarse para lograr un objetivo. El objetivo de la fase de análisis de requisitos es capturar el detalle de cada requisito y asegurarse de que todos comprendan el alcance del trabajo y cómo se va a cumplir cada requisito.
- Diseño. Durante la fase de diseño, los desarrolladores y arquitectos técnicos inician el diseño de alto nivel del software y el sistema para poder cumplir con cada requisito.
    - Desarrollo. Se implementa la solución de acorde al diseño y los requisitos
- Pruebas. Es la última fase del ciclo de vida del desarrollo del software antes de que el software se entregue los clientes. Durante las pruebas, los probadores experimentados comienzan a probar el sistema contra los requisitos.
- Despliegue. Una vez que el software se haya probado por completo y no haya problemas, es hora de implementarlo en la producción donde los clientes pueden usar el sistema.

1.2.2. Modelos de ciclo de vida

El modelo de ciclo de vida establece el flujo del proceso de desarrollo de software, indica el seguimiento de las etapas a realizar, el proceso puede ser lineal, iterativo o evolutivo. Hay varios modelos de ciclo de vida de desarrollo de software, entre los más destacados están: Cascada, desarrollo evolutivo, iterativo, basado en componentes y desarrollo ágil, a su vez podemos ver las características de cada uno en la Tabla 3. (Rea, 2017)

Tabla 3: Modelos de ciclo de vida del software

MODELO	CARACTERÍSTICAS
Cascada	- El conocimiento total de los requerimientos de un sistema debe ser obtenido antes de su desarrollo. - Cada fase es aceptada por los usuarios antes de avanzar con la siguiente fase. No es posible regresar a fases anteriores. - El desarrollo del sistema puede tomar mucho tiempo. - Permite obtener un sistema comprensible, documentado, funcional y bien integrado. - Sin embargo, los requerimientos pueden cambiar antes de que el sistema se finalice.
Desarrollo Evolutivo	- Permite el desarrollo de una implementación inicial del sistema, revisarla con el usuario y refinarla las veces que sean necesarias hasta su finalización. - Permite obtener retroalimentación del usuario en cada refinamiento. - La especificación de requerimientos puede desarrollarse en forma creciente. - El usuario obtiene una idea del sistema antes de su finalización.
Iterativo	- El ciclo de vida se compone de varias iteraciones. - Cada iteración es un mini proyecto que tiene todas las fases de un ciclo de vida. - Las iteraciones pueden ser en serie o en paralelo, pero todas son integradas en la versión final del producto software
Desarrollo Ágil	- Pone mayor énfasis en los resultados de la Ingeniería de Software (construcción y entrega), más que en el proceso de obtener esos resultados (análisis, diseño, desarrollo, implementación y administración).

	- Los requerimientos del software pueden ir evolucionando en el tiempo de acuerdo con la necesidad. - Se planifica la entrega del software por partes funcionales, de acuerdo con la priorización del usuario. - Según el (Manifiesto Ágil, 2001), se prefiere el software funcionando sobre la documentación extensiva, los individuos e interacciones sobre procesos y herramientas, la colaboración con el cliente sobre negociación contractual y la respuesta ante el cambio sobre seguir un plan.

Fuente (Rea, 2017)

1.3. Metodologías ágiles de desarrollo

Como lo menciona (Ludeña, 2018), “Los métodos ágiles son aquellos que permiten adaptar la forma de trabajo al contexto y naturaleza de un proyecto, basándose en la flexibilidad y la rapidez, teniendo en cuenta las exigencias del mercado y los clientes. Los métodos ágiles nacen en contraposición a lo que representaban los métodos tradicionales”

Ante la constante demanda de desarrollo de software para pequeñas organizaciones, la necesidad de soluciones de bajo costo llevó al crecimiento de métodos más simples y rápidos que ayuden a la gestión de proyectos de software orientando que el producto sea funcional; el uso de prototipos, métodos, procedimientos y normas evoluciona al uso del concepto de metodología del proceso de desarrollo de software, destacando el uso de metodologías ágiles o ligeras. Una de ellas fue XP que simplificó muchas áreas de la ingeniería del software como la forma de levantamiento y recopilación de requerimientos, y las pruebas de confiabilidad. Los productos de software muy grandes aún utilizan metodologías rígidas y muy documentadas; sin embargo, los sistemas más pequeños tienen un enfoque alternativo simple y rápido de gestionar todo el ciclo de desarrollo del software (Chamorro, 2018).

1.4. Gestión de proyectos de software

1.4.1. Definición

En el pasado, muchos de los proyectos de software fallaron antes de que se completara la etapa, se debe a la falta de planificación y falta de proyecto técnicas y herramientas de gestión. Buena gestión de proyectos siempre conducirá al éxito del software. La gestión de proyectos de software incluye las áreas de conocimiento, herramientas y técnicas. Consta de diez áreas de conocimiento que tienen cuatro núcleos y seis son de apoyo. Las áreas de conocimiento básicas so proyectos gestión de costes, gestión de costes del proyecto, alcance del proyecto gestión y gestión de costes del proyecto y apoyo las áreas de conocimiento son la gestión de recursos humanos del proyecto, gestión de adquisiciones, com. proyecto. administración, gestión de grupos de interés, gestión de riesgos y proyectos, gestión de la integración que integra todo el conocimiento áreas que tiene 5 procesos como iniciar, planificar, ejecutar, monitorear y controlar. Si el software se desarrolla siguiendo todas las herramientas y técnicas específicas de PM que definitivamente conducirán al éxito (Hayat et al., 2019).

El conjunto de herramientas que permite al personal encargado desarrollar habilidades para afrontar los retos, contingencias, adversidades y las nuevas situaciones que impone el entorno de cambio continuo se denomina Gestión de Proyectos. El proyecto es un conjunto de actividades planificadas, ejecutadas y monitoreadas que generan un resultado único, contemplando a un equipo y entregando un resultado posiblemente en forma de productos o servicios. (Almeida & Correia, 2019)

También denominada administración de proyectos, la gestión de proyectos se considera a la aplicación de habilidades, conocimientos y técnicas en la definición de actividades enfocadas para alcanzar una serie de objetivos planteados.

1.4.2. Scrum en la gestión de proyectos

El desarrollo de software es un proceso colaborativo que necesita reunir experiencia en el área, habilidades tecnológicas y conocimientos dentro del proceso. Así mismo constituye la recopilación de requisitos, diseño, desarrollo, pruebas, implementación y mantenimiento. Es casi común que los miembros del equipo tengan todos los conocimientos necesarios para desarrollar estas actividades, por lo que es imprescindible priorizar la comunicación, compartir los conocimientos y así evitar la pérdida de conocimientos con la partida de u trabajador experimentado (Cabral et al., 2014)

La metodología ágil se adapta a los cambios de requisitos en el futuro, y tiene tal flexibilidad que puede manejar el costo, el alcance, calidad del software según las necesidades del cliente. Scrum es el marco de metodología ágil ya que se enfoca en el día a

día gestión de proyectos y es más ampliamente adoptar el método de gestión de proyectos ágil. usamos el marco de scrum para averiguar el impacto de la metodología ágil en la gestión de proyectos de software en la industria del software de Pakistán, ya que el uso de scrum aumenta considerablemente día a día y muchos proyectos de software se desarrollan mediante el uso de scrum. En proyectos distribuidos globalmente, existe un interés creciente y una demanda de literatura que utiliza prácticas de scrum para comprender el estudio empírico (Hayat et al., 2019)

1.4.3. GitLab en la gestión de proyectos

GitLab es una aplicación para el ciclo de vida del desarrollo de software. Está diseñado para gestionar la totalidad de un proyecto de software, incluida la gestión del código fuente, la gestión de proyectos, el control de calidad y la gestión de versiones. GitLab y su cadena de herramientas pueden soportar todo el ciclo de “DevOps”.

La gestión de proyectos es fundamental para GitLab. El seguimiento de problemas está integrado con muchas de las otras herramientas que ofrece GitLab. Es posible hacer referencia a confirmaciones, realizar solicitudes de combinación o utilizar problemas para analizar el progreso. También se pueden utilizar rastreadores de problemas externos, como Redmine, Bugzilla y muchos otros.

Hay muchas formas de comunicarse, compartir ideas y codificar. La documentación se puede crear usando la wiki incorporada, si no se desea almacenarla en el repositorio de código. Se puede usar un IDE web para programar directamente usando la interfaz web o para hacer arreglos rápidos. Existen herramientas de análisis y seguimiento del tiempo.

En otras palabras, GitLab contiene muchas herramientas para crear, construir, monitorear y administrar un proyecto. Y esto se hace de una manera en la que el equipo de operaciones no se ve obligado a utilizar las herramientas integradas (Mikko Dittmer, 2019).

CAPÍTULO II

Desarrollo

2.1. Marco de trabajo Scrum

2.1.1. Definición

Según el portal *ProyectosAgiles.org*, determinan a la *Metodología Ágil Scrum* como: un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para Trabajar Colaborativamente, en Equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos (Renato, 2016).

2.1.2. Propósito de la Guía de Scrum

Scrum es un marco de trabajo para desarrollar, entregar y mantener productos complejos, Esta guía contiene un conjunto de reglas que abarcan los roles, eventos y artefactos de Scrum (Schwaber & Sutherland, 2017)

2.1.3. Teoría de Scrum

Scrum está basado en la teoría de control empírico o empirismo. EL empirismo afirma que el conocimiento proviene de la experiencia y se basa en decisiones conocidas.

Existen tres pilares que sustentan la implementación del control de procesos empírico: transparencia, inspección y adaptación.

Transparencia

Los temas relevantes que intervienen en el proceso deben ser sumamente claros y visibles para el personal encargado del resultado

Inspección

Los usuarios de Scrum siempre deben verificar los artefactos de Scrum y avanzar hacia un objetivo para eliminar los cambios no deseados.

Adaptación

Si se verifica que uno o más incidentes de un proceso se sobrepasan del límite establecido, dicho proceso o actividad deberá ajustarse lo más pronto posible para minimizar desviaciones importantes. (Schwaber & Sutherland, 2017)

2.1.4. El Equipo Scrum (Scrum Team)

El Equipo Scrum esta conformado por el Dueño del Producto (Product Owner), el Equipo de Desarrollo (Development Team) y el Scrum Master.

2.1.4.1. El Dueño de Producto (Product Owner)

El Dueño de Producto es el encargado de maximizar el valor del trabajo generado del Equpo de Desarrollo.

El Dueño de Producto es el único encargado de administrar la Lista del Producto (Product Backlog). Este proceso constituye:

- Expresar claramente los elementos de la Lista del Producto
- Ordenar los elementos en la Lista del Producto para alcanzar los objetivos
- Optimizar el valor del trabajo realizado por el Equipo de Desarrollo
- Verificar que la Lista del Producto es clara y transparente para todos
- Verificar que el Equipo de Desarrollo comprende los elementos de la Lista del Producto

(Schwaber & Sutherland, 2017)

2.1.4.2. El Equipo de Desarrollo (Development Team)

El Equipo de Desarrollo lo conforman los encargados que trabajan en la entrega de un incremento de producto en estado de “terminado” que temporalmente pueda lanzarse a producción al final de cada sprint. Un incremento “terminado” deberá estar en la revisión del sprint.

El Equipo de desarrollo tiene las siguientes características:

- Son autogestionados. Nadie podrá dar indicaciones de cómo hacer su trabajo con respecto al desarrollo de los incrementos funcionales del producto.
- Son multifuncionales, es decir que poseen todas las habilidades para lograr el incremento de producto.

- Scrum no reconoce títulos para los integrantes del equipo de desarrollo independientemente de la actividad que realice cada uno.
  (Schwaber & Sutherland, 2017)

2.1.4.3. El Scrum master

El Scrum master es el responsable de llevar a cabo al marco de trabajo Scrum de acuerdo a la guía de Scrum. El Scrum master es el encargado de que todos los miembros entiendan la teoría, proceso, reglas y valores de Scrum.

El Scrum master es quien lidera y está al servicio del Equipo Scrum, es el encargado directo de ayudar y apoyar constantemente a todos en modificar, reestructurar, distribuir las interacciones para lograr maximizar el valor creado por el Equipo Scrum.

El Scrum master es el responsable directo de brindar servicio y asistencia al Dueño del producto, Equipo de desarrollo y a la Organización. (Schwaber & Sutherland, 2017)

2.1.5. Eventos de Scrum

Los eventos en Scrum son bloques de tiempo, los cuales tienen una duración máxima para ejecutarse (Schwaber & Sutherland, 2017).

2.1.5.1. El Sprint

EL Sprint es el evento principal de Scrum, consiste en un periodo de tiempo que puede tener una duración de un mes o menos dentro del cual se crea un incremento del producto entregable o “terminado”.

Todo Sprint tiene trazada una meta de lo que se realizará, posee un conjunto de tareas que definirá su desarrollo, el trabajo del equipo y el incremento de producto resultante.

2.1.5.2. Planificación de Sprint (Sprint Planning)

El trabajo que será desarrollado durante el Sprint se planifica en la Planificación de Sprint.

2.1.5.3. Objetivo del Sprint (Sprint Goal)

El Objetivo del Sprint es la meta trazada para el Sprint que esta asignado por el Product Backlog. Los elementos del Product Backlog seleccionados definen el margen que puede ser el objetivo del Sprint.

2.1.5.4. Scrum Diario (Daily Scrum)

El Scrum diario es una reunión dedicada solo para el Developer Team con una duración de 15 minutos. El Daily Scrum se cumple cada día del sprint y en el cual el Developer Team planifica el trabajo para las siguientes 24 horas.

2.1.5.5. Revision de Sprint (Sprint Review)

Cuando termina el sprint se lleva a cabo una revisión de sprint para analizar el incremento del producto y reorganizar el Product backlog en caso se lo requiera.

La finalidad de la revisión de sprint es un product backlog revisado que define los elementos del producto backlog posible para el siguiente sprint.

2.1.5.6. Retrospectiva de Sprint (Sprint Retrospective)

La retrospectiva de sprint se da lugar después de la revisión de sprint y antes de la siguiente planificación de sprint.

La restrospectiva de sprint es una autoevaluación para el equipo scrum y crear un plan de mejoras que serán tomadas en cuenta durante el siguiente sprint.

2.1.6. Artefactos de Scrum

2.1.6.1. Lista de Producto (Product Backlog)

El Product Backlog es una lista ordenada de todo lo que se conoce que es necesario en el producto. Es la única fuente de requisitos del producto. El Product owner es el responsable del Product backlog, incluyendo su contenido, disponibilidad y orden.

El Product Backlog es dinámico, cambia en todo momento. El Product Backlog enumera todas las características y funcionalidades.

2.1.6.2. Lista de Pendientes del Sprint (Sprint Backlog)

El Sprint Backlog consta de una lista de actividades (comentarios de codificación, pruebas unitarias, prototipos de diseño, documentos, etc.) que debe cumplir cada una de las historias de usuario que forman parte del Sprint Backlog.

2.2. Plataforma GitLab

2.2.1. Funcionalidades de GitLab

En GitLab podemos gestionar principalmente proyectos, grupos y fragmentos. Los proyectos son el eje central del sistema, básicamente repositorios de software gestionados en su totalidad por la plataforma GitLab.

2.2.2. Gestor de código fuente

En este bloque intervienen control de versiones como Git, donde los usuarios no solo revisan la última aplicación instantánea de los archivos; más bien, en este versionamiento se reflejan todos los archivos en el repositorio, incluido su historial completo. Por lo tanto, si algún servidor muere y estos sistemas colaboran a través de ese servidor, cualquiera de los repositorios de los clientes se puede copiar en el servidor para restaurarlo, este clon es realmente una copia de seguridad completa de todos los datos.(Soria José, 2021)

2.2.3. Gestión del proyecto

GitLab permite la gestión de proyectos ágiles, desde el seguimiento básico de problemas hasta la gestión de proyectos de estilo Scrum y Kanban. Ya sea que simplemente esté rastreando algunos problemas o administrando el ciclo de vida completo de DevOps en un equipo de desarrolladores, GitLab tiene a su equipo cubierto.

- Planifique, asigne y realice un seguimiento de los problema**s**
- Organizar el trabajo con etiquetas, iteraciones e hitos
- Visualizar el trabajo con tableros
- Correlacionar el trabajo con la salida mediante solicitudes de combinación

2.2.4. Fase de Administración

2.2.4.1. Autenticación

GItlab te ofrece varias formas de autenticación tales como:

- Username y Password
- Two factor authentication(2FA)

- SSH Key

La Llave SSH o SSH Key es el método más usado para autenticarse con GitLab y poder subir nuestro código de nuestra computadora a través de los comandos de Git.

2.2.4.2. Grupos

Los grupos te permiten compartir recursos entre varios miembros del equipo y organizar la forma en la que trabajas.

- Agrupar proyectos: dónde va a vivir nuestro código y los recursos asociados.
- Otorgar permisos de acceso: qué usuarios de qué equipo van a poder acceder a los recursos de la compañía.
- Compartir recursos: Si se tiene cluster de Kubernetes, templates o runners para correr el CI se puede compartir entre varios grupos.

Los grupos se utilizan en Gitlab a través de los Namespaces que nos dan una url única para nuestro usuario, grupo y subgrupo.

2.2.4.3. Autorización

Existen diferentes formas de autorizar un usuario dentro de un grupo y estos mismos modelos se utilizan para los proyectos:

- Guest: Es Read-only, solo tiene permisos de lecturas. No puede modificar código, crear o comentar issues.
- Reporter: Solo puede crear y comentar en los issues. No puede añadir código.
- Developer: Puede añadir código, también da acceso a los Pipelines de CI, branchs y más, pero no da la capacidad de agregar nuevos miembros.
- Owner / Maintainer: Eres Owner cuando creas un proyecto y Maintainer cuando alguien más te da permisos para administrar ese proyecto.

2.2.4.4. Proyectos

Los proyectos tienen tres componentes fundamentales:

- Issue tracker: No es uno genérico para cualquier compañía, es específico para equipos que desarrollan software y adentro encontrarás conceptos como milestone, estimados de tiempo y más.

- Code repository: Es el lugar centrar que nos va a permitir compartir y colaborar alrededor del código.

- Gitlab CI: Nos muestra la posibilidad de automatización con la inclusión de Continuous Integration.

2.2.5. Fase de planificación

2.2.5.1. Issues

Los issues son el punto donde inicia una conversación sobre el código. Son útiles para definir cualquier problema que se encuentra en el software y darle seguimiento, vendrían a ser como las historias de usuario en Scrum.

Los issues permiten:

- Discutir la implementación de una nueva idea.
- Sugerir propuestas de features.
- Hacer preguntas.
- Reportar bugs y fallos.
- Obtener soporte.
- Planear nuevas implementaciones de código

En la Fig. 5 tenemos los campos requeridos para añadir un nuevo Issue

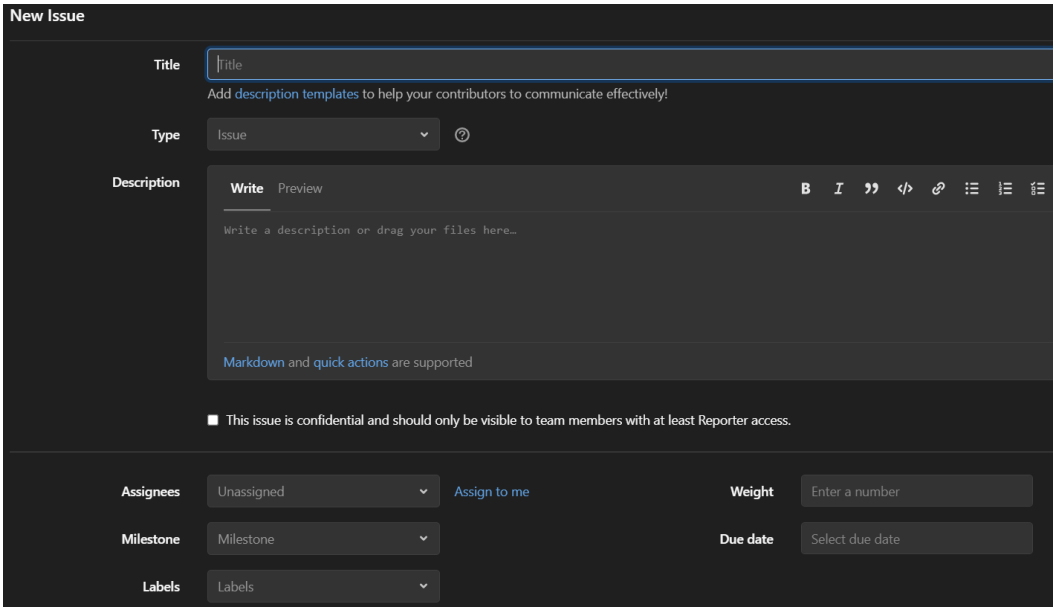


Figura 5: Creación de un issue
Fuente: Propia

2.2.5.2. Labels (Etiquetas)

El siguiente paso es clasificar los issues basados en etiquetas, suele salirse de control la forma en la que se reportan y una manera sencilla de organizar es con etiquetas.

Las etiquetas nos permiten:

- Categorizar issues o merge request con títulos descriptivos.
- Filtrar y buscar en Gitlab
- Seguir temas a través de notificaciones.

En la Fig.6 nos muestra los campos para crear un label.

Figura 6: Crear un label

Fuente: Propia

2.2.5.3. Milestones (Hitos)

Los milestones son una forma de realizar un seguimiento de los issues con el fin de lograr el objetivo en un peródo de tiempo. Ver Fig. 7 para crear un milestone.

Los milestones se pueden usar como sprints ágiles para poder dar seguimiento a todos los issues, para ello se deberá:

- Establecer la fecha de inicio y la fecha de vencimiento para representar el inicio y el final de un sprint ágil.
- Establecer el título del milestone como nombre del respectivo sprint, es decir “Sprint 1 (fecha)”
- Agregar uno o varios issues al sprint asociado desde la lista lateral derecha de issues

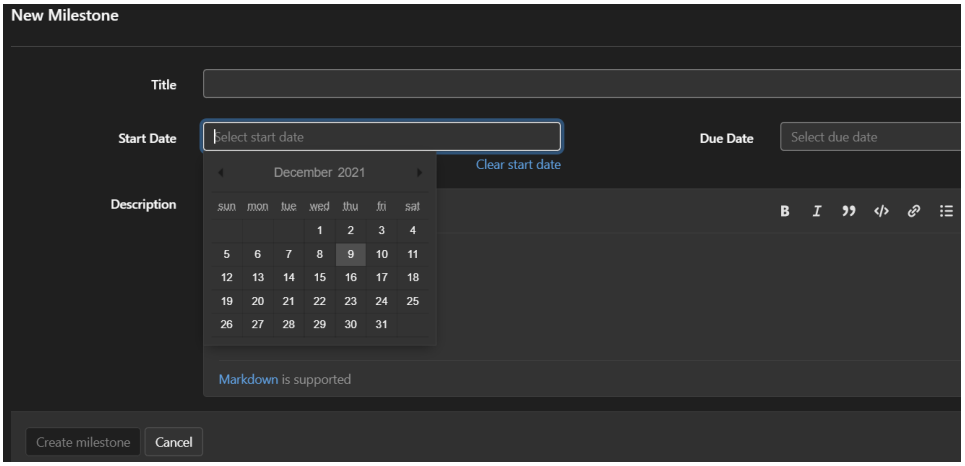


Figura 7: Crear nuevo milestone

Fuente: Propia

2.2.5.4. Boards

Los tableros de issues son una herramienta de gestión de proyectos de software que se utiliza para planificar, organizar y visualizar un flujo de trabajo para una versión de característica o producto. Se los puede utilizar como un tablero Kanban o Scrum. La Fig. 8 nos indica los tableros con las issues asociadas.

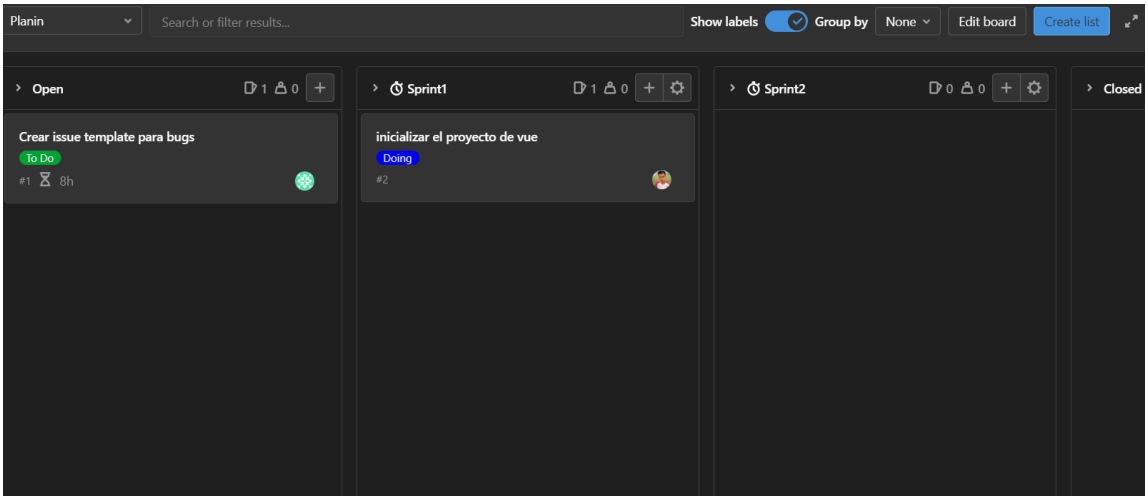


Figura 8: Board con milestones

Fuente: Propia

2.2.6. Verificación

2.2.6.1. Inicialización del repositorio

En este apartado tiene la finalidad de dar inicio a la creación de la aplicación (proyecto) en la cual se agregará el código fuente a un repositorio, crear merge request para la integración de código y utilizar CI/CD para generar la aplicación.

Un repositorio es el lugar donde se almacena el código y se realizan cambios en él. Los cambios se controlan con el control de versiones (Git). Cada proyecto contiene un repositorio. En la Fig. 9 se presenta la plantilla de comandos para inicialización de un proyecto en GitLab a través de comandos de git.

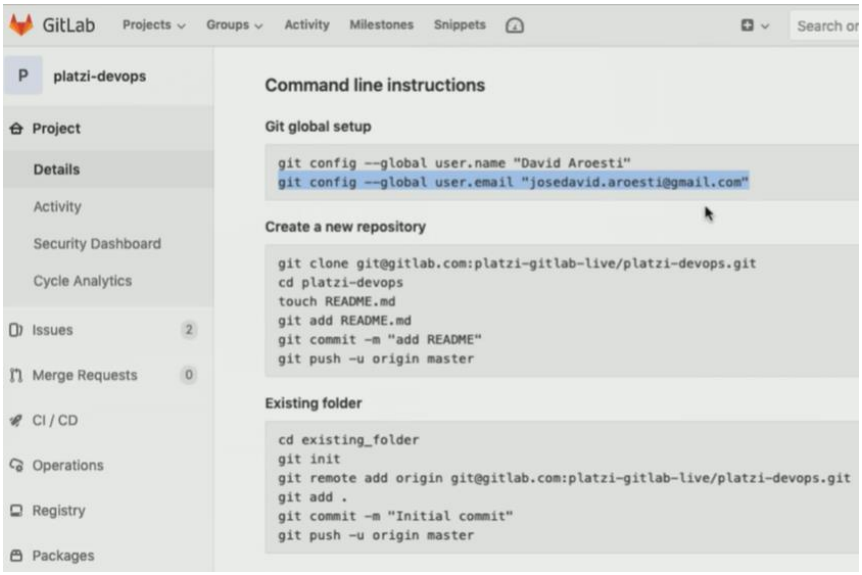


Figura 9: Instrucciones y comandos para inicializar un repositorio de Git.
Fuente: Propia

2.2.6.2. Merge Request

Son como las Pull Request de GitHub en donde te permiten controlar todas las solicitudes de combinación o unión de código de distintas ramas o forks. Es muy importante que los merges se resuelvan mediante la interfaz gráfica, ya que nos ofrece muchas posibilidades interesantes, como automatización de tests, la posibilidad de revisión de los cambios por parte de componentes del equipo, implementar diversas políticas de control sobre el código del proyecto, etc.

2.2.6.3. CI/CD

Es una de las maravillas que dispone GitLab, una herramienta sencilla y muy útil para los procesos de integración continúa y despliegue continúo es así como existen muchas herramientas que se puede integrar para automatizar los procesos y llegar a crea flujos de trabajo completamente automatizados.

2.3. Elaboración de la guía metodológica

2.3.1. Introducción

La guía metodológica de procedimientos y estrategias para aplicar Scrum con GitLab en el desarrollo de proyectos de software desarrollada a continuación es el producto de la investigación de todos aquellos conceptos definidos a lo largo del estudio realizado.

La presente guía metodológica contiene la implementación del marco de trabajo Scrum para la gestión de proyectos de software utilizando la herramienta GitLab.

Este documento describe los pasos prácticos para la gestión de proyectos de desarrollo de software, el mismo que contiene las actividades principales establecidas por Scrum realizadas por medio de la herramienta GitLab.

Esta propuesta consiste en una guía práctica para la aplicación de esta metodología de manera que se pueda optimizar procesos y se tenga la oportunidad de compartir información entre los miembros del equipo.

El contenido de la presente contiene pasos a seguir desde el inicio del proyecto hasta su fecha de finalización y entrega del producto, en base a toda la investigación efectuada se pudo evidenciar que una guía realista sería una herramienta muy útil para cualquier persona que necesite empezar con Scrum y GitLab en cualquier proyecto, siendo de apoyo didáctico que beneficiara no solo al proyecto sino a todo el equipo de trabajo.

2.3.2. Objetivos

Establecer una guía metodológica acerca del uso de Scrum con la herramienta GitLab.

2.3.3. Alcance

Tener un instrumento que sirva como un manual de referencia para que los implicados y/o interesados (analistas, desarrolladores, diseñadores, etc.) que quieran gestionar sus proyectos mediante Scrum con GitLab.

2.3.4. Terminología

Se presenta un listado de términos presentes en GitLab y a su vez equivalente a los componentes de Scrum como se aprecia en la Tabla 4.

Tabla 4. Terminología de la guía

Término	Significado
Issues	Historia de ususario, tarea, feature, bug
Weight	Puntos de esfuerzo o puntos del Issue
Time tracking	Estimacion, duración del Issue
Milestones	Sprints, versiones
Assignees	Encargados o responsables de una actividad
Time tracking	Seguimiento del tiempo, tiempo estimado para un issue
Tags	Etiquetas de git para identificar versiones

Fuente: Propia

2.3.5. Responsables

Autor: Sr. Erick Vásconez

Tutor: MSc. Mauricio Rea

2.3.6. Descripción de las actividades

2.3.6.1. Fase 1. Pre-Juego

1.1 Inicio del proyecto

Crear proyecto con el nombre a convenir, es importante crear el proyecto en modo **Público** para utilizar todos los beneficios gratuitos de GitLab

Subir código fuente al repositorio de GitLab

Una vez creado el proyecto, se procede a inicializar el proyecto y el repositorio de código del proyecto con los comandos git que nos indica en GitLab. En la Fig.10

Git global setup

```
git config --global user.name "Erick Alexander Vásconez Endara"
git config --global user.email "eavasconeze@utn.edu.ec"
```

Create a new repository

```
git clone git@gitlab.com:erickcr91496/testproject.git
cd testproject
git switch -c main
touch README.md
git add README.md
git commit -m "add README"
git push -u origin main
```

Push an existing folder

```
cd existing_folder
git init --initial-branch=main
git remote add origin git@gitlab.com:erickcr91496/testproject.git
git add .
git commit -m "Initial commit"
git push -u origin main
```

Push an existing Git repository

```
cd existing_repo
git remote rename origin old-origin
git remote add origin git@gitlab.com:erickcr91496/testproject.git
git push -u origin --all
git push -u origin --tags
```

Figura 10 Configuraciones inciales del proyecto

Fuente: Propia

1.2 Definición de Roles Scrum

Para dar inicio al uso del marco de trabajo de Scrum se debe definir los roles que solicita Scrum.

Los roles de Scrum son: Product Owner, Scrum Master y Development Team.

Agregar a los miembros del equipo al proyecto

Dentro de la información del Proyecto en Members agregamos a los miembros del mismo con su rol respectivo. En la Fig. 11 se visualiza la ventana de agregar miembros al equipo.

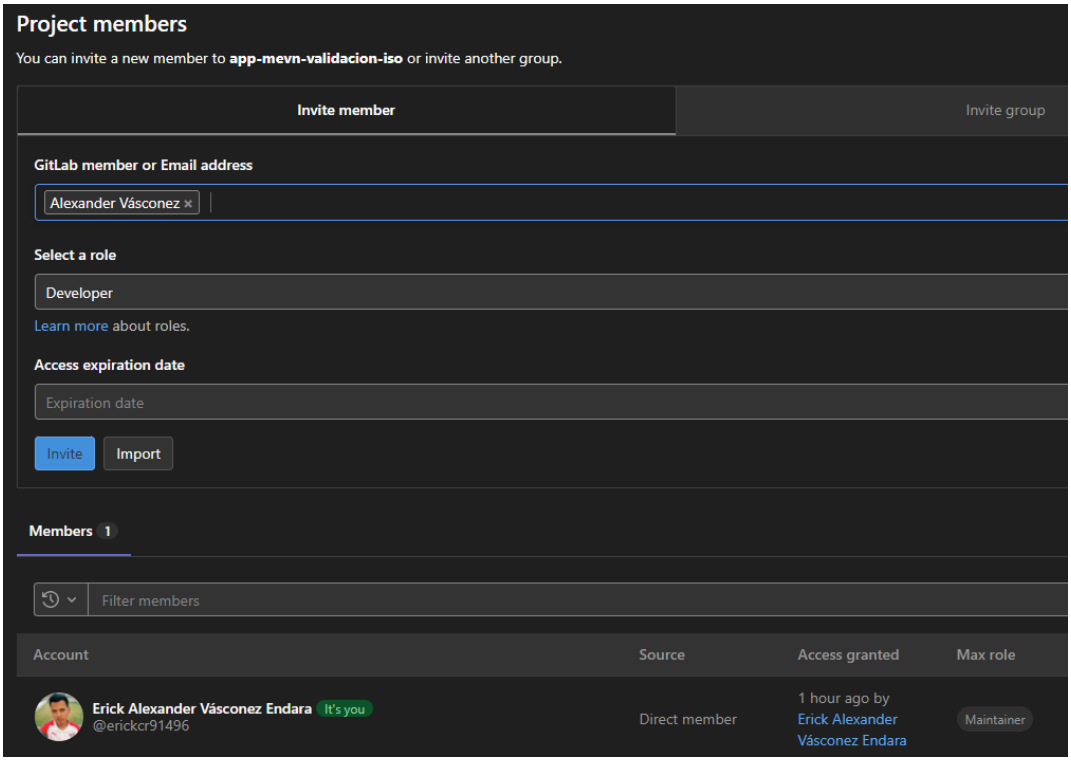


Figura 11 Miembros del equipo en GitLab

Fuente: Propia

1.3 Definición del Product Backlog

En este apartado se procede a definir la lista ordenada de requisitos o funcionalidades necesarias para la construcción del producto final.

El Product Backlog enumera todas las características, funcionalidades, requisitos, mejoras y correcciones a realizar sobre el producto para entregas futuras. Es una lista ordenada normalmente por prioridad y estimación de las historias de usuario.

La plantilla general para realizar el Product Backlog se presenta en la Tabla 5:

Tabla 5 Definición del Product Backlog

ID	Historias de Usuario	Prioridad	Estimación	Descripción
# Orden por prioridad	Título de historia de usuario	Puntos de historia	Tiempo/Duración	Resumen del objetivo de la historia de usuario

Fuente: Propia

Creación de Issues del Product backlog

El Product Backlog en GitLab es una lista de issues ordenados y priorizados por medio de etiquetas o labels.

Historias de Usuario = Issue

Prioridad = Weight (Peso, valor)

Estimacion = Time traking

Para la creación de una Issue seguiremos los pasos descritos a continuación junto con todos los campos que solicita la metodología Scrum configurados en GitLab.

Dirigirse al apartado de Issues y hacemos click en **New issue**.

Una vez abierto el formulario se solicitará llenar los siguientes campos, ver Fig. 12

- Title: Título o nombre del Issue
- Type: Seleccionar si es de tipo Issue o Incident (bug)
- Description: Descripción del Issue, criterios de aceptación, etc
- Assignees: Delegaremos el Issue o tarea a un miembro del equipo
- Milestone: Seleccionamos el Sprint correspondiente, en caso de no tener un milestone procedemos a crear uno en el **paso 1.4.**
- Labels: Etiquetas que podemos definir a cada Issue para identificar
- Weight: Definimos un rango de puntuación y se coloca el peso de acuerdo a la prioridad del issue.
- Due date: Definimos la fecha de finalización del Issue.

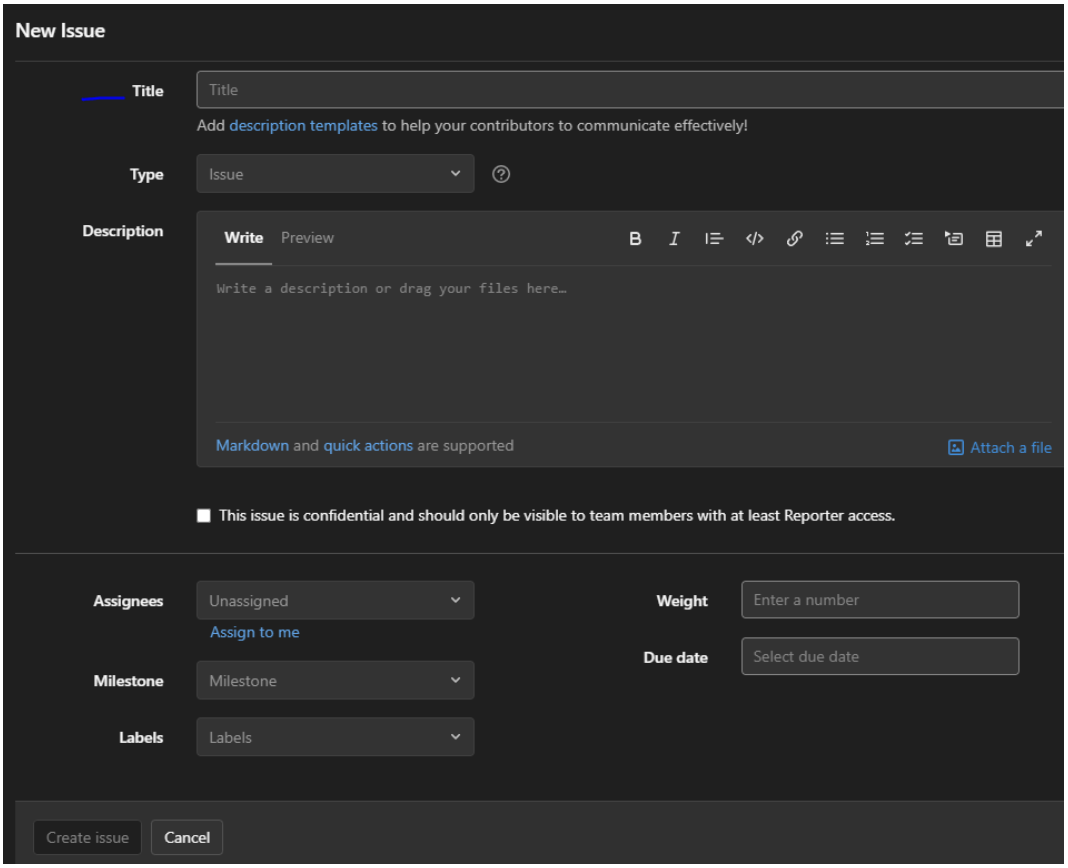


Figura 12 Campos de un Issue

Fuente: Propia

1.4 Creación de los Milestones o Sprints del proyecto

Dentro de la pestaña Milestones hacemos click en New Milestone y procedemos a llegar los siguientes campos. Ver Fig 13.

- Title: Definimos el nombre del Sprint, en nuestro caso Sprint1,…Sprint N
- Start Date: Fecha de inicio del Sprint
- Due Date: Fecha fin del Sprint
- Description: Una breve descripción del objetivo del Sprint

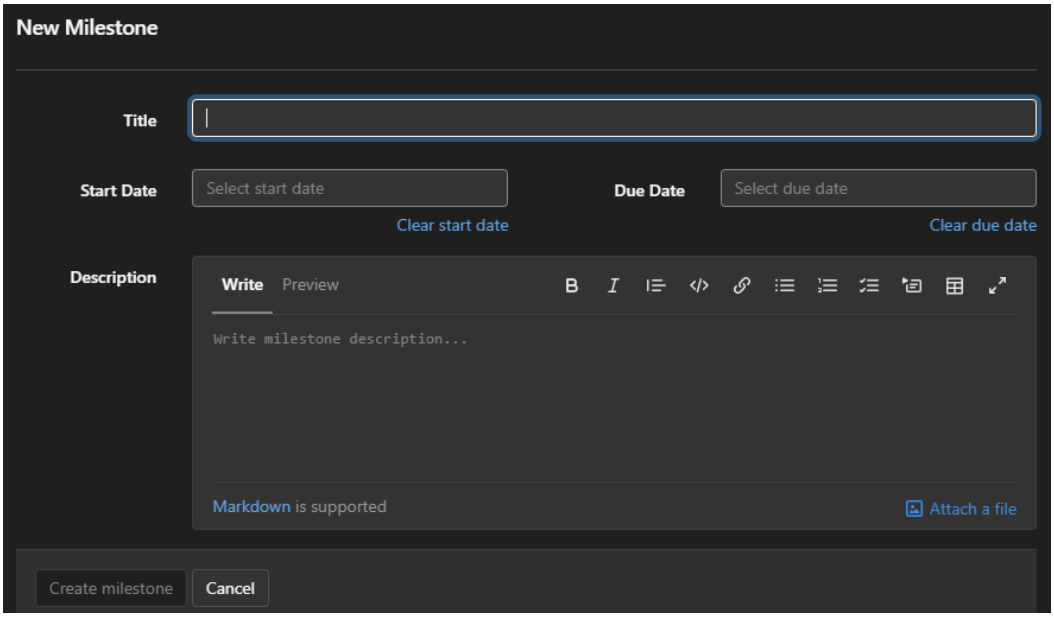


Figura 13 Pantalla de creación de un Milestone o Sprint

Fuente: Propia

1.5 Definición de historias de usuario

En las historias de usuario se procede a especificar los requerimientos que el usuario necesita realizar dentro del sistema.

1. La plantilla y campos para una historia de usuario ver Tabla 6.

Tabla 6 Plantilla de historia de usuario

HISTORIA DE USUARIO	
ID: #	**Usuario:**
Nombre de historia:	
Prioridad:	**Estimación:**
Descripción:	
Criterios de aceptación:	

Fuente: Propia

Una historia de usuario simplemente es un Issue con los campos descritos anteriormente en la Fig.10 a excepción de los milestones, los cuales serán asignados en el Sprint Backlog.

En la Fig.14 se puede ver un ejemplo de un Issue creado.

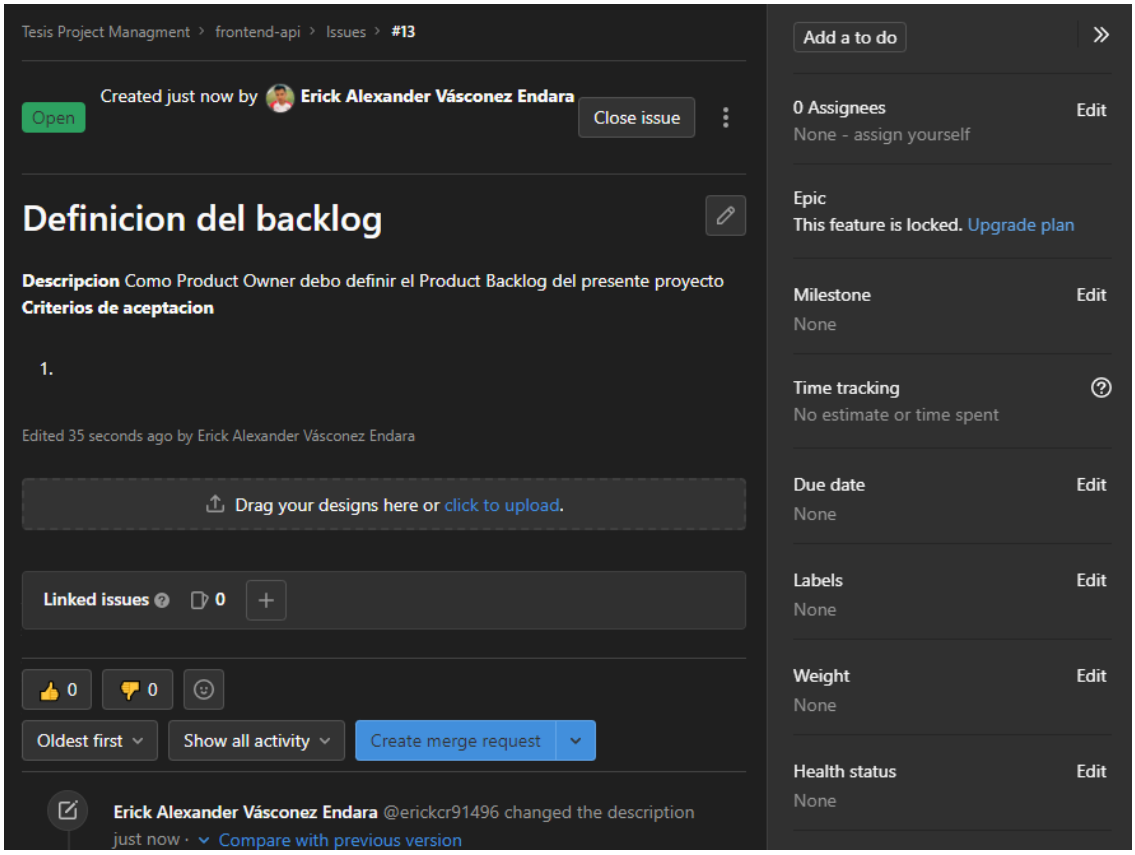


Figura 14 Carácteristicas de un Issue

Fuente: Propia

Lista de tareas de issue

1. Para describir las actividades o tareas del issue, se procede a escoger la opción de **task list** en la barra de herramientas de la caja de texto del issue correspondiente. Ver Fig.15.

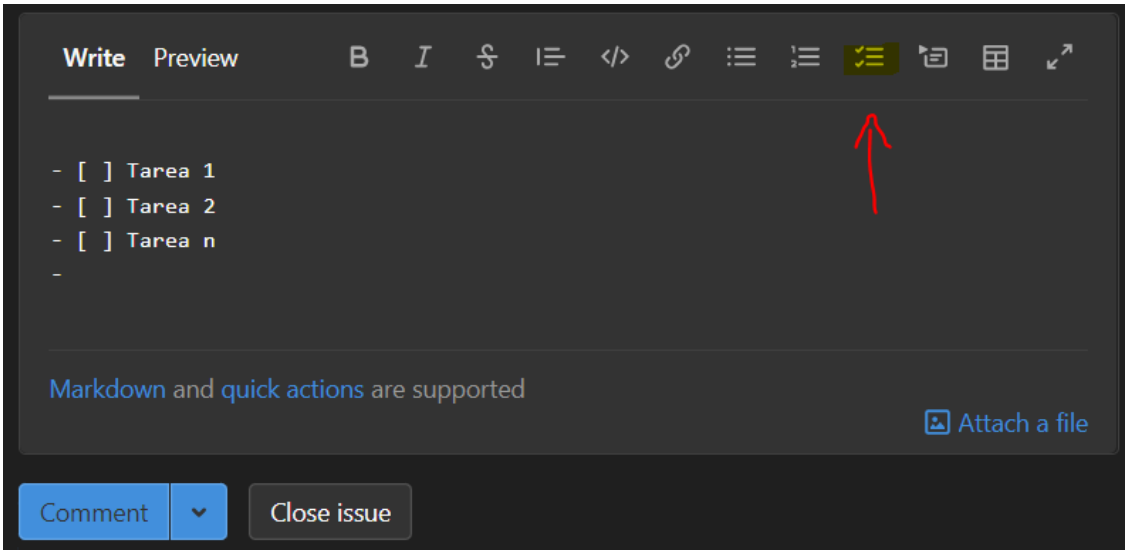


Figura 15 Listado de tareas de un issue

Fuente: Propia

2. Una vez descritas las tareas, presionamos en **Comment y** se visualizará como se ve en la Fig. 16

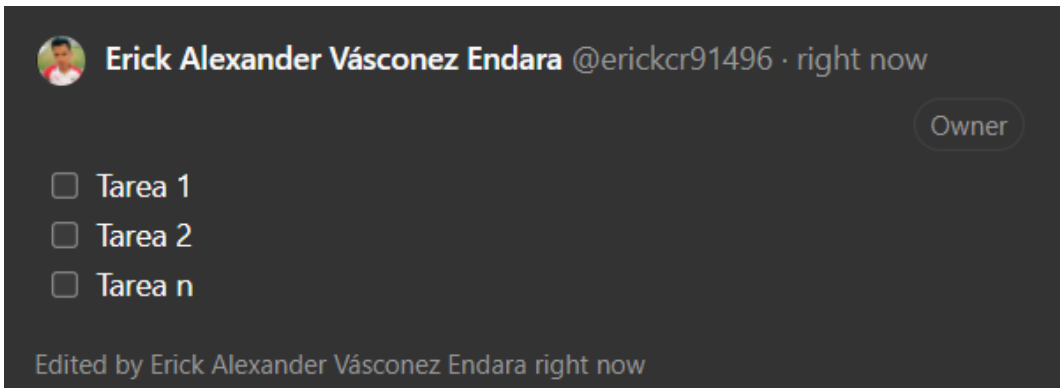


Figura 16 Agregar tareas de issue

Fuente: Propia

3. Cada tarea que se ha completado del issue se procederá a marcar con visto la casilla de la tarea realizada.

Time tracking (Estimación)

Con el Time tracking se puede realizar un seguimiento de las estimaciones y el tiempo dedicado a los Issues.

1. *Usos*

Esta funcionalidad se aplica para las siguientes tareas:

- Registrar el tiempo dedicado a trabajar en un Issue
- Agregar o actualizar una estimación del tiempo total para completar un Issue

Los campos sobre el Time tracking aparecen en la barra lateral del Issue. Ver Fig.17

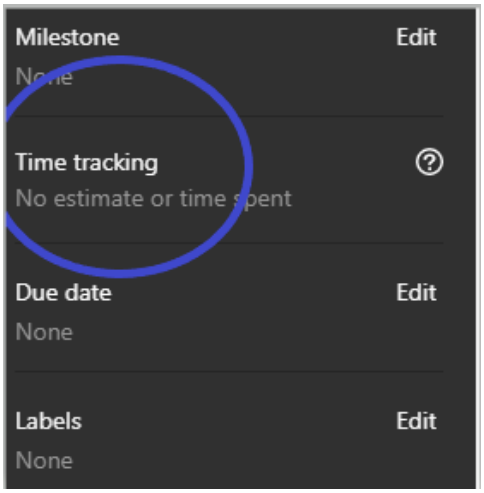


Figura 17 Time tracking de un issue
Fuente: Propia

2. Insertar datos

Para registrar el tiempo el time tracking utiliza dos acciones rápidas: /estimate y /spend

Introducir una estimación

Dentro del comentario se escribe el término /estimate seguido del tiempo en unidades de horas, días, semanas, meses y minutos.

Ejemplo:

/estimate 1mo 2w 3d 4h 5m

En la Fig.18 se visualiza el panel de estimación asignado para un Issue.

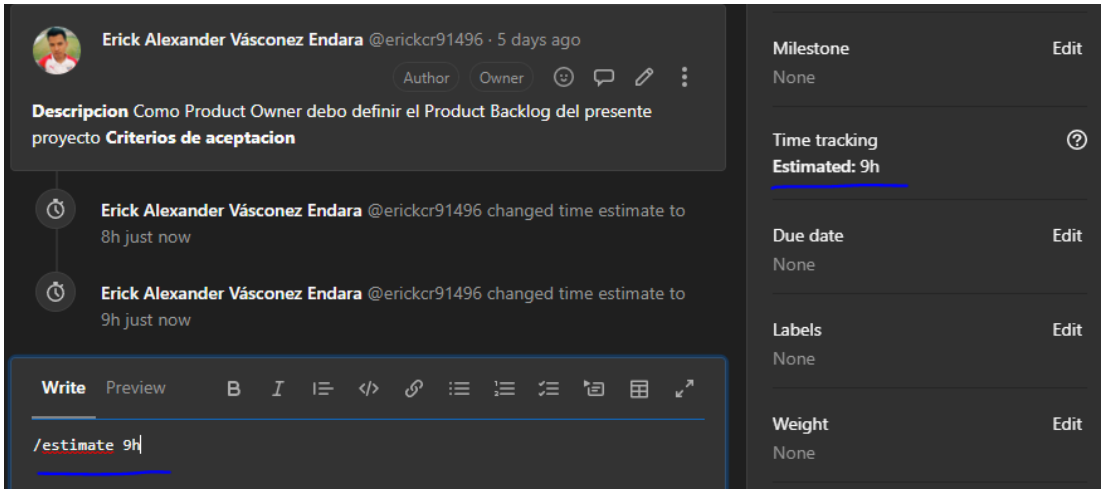


Figura 18 Estimación asignada

Fuente: Propia

Introducir el tiempo empleado o dedicado (Spend)

Dentro del comentario escribir el término /spend seguido de la unidad de tiempo al igual que en el paso 1.

Ejemplo: /spend 1mo 2w 3d 4h 5m

En la Fig.19 se visualiza el panel de estimación con la barra de progreso del spend.

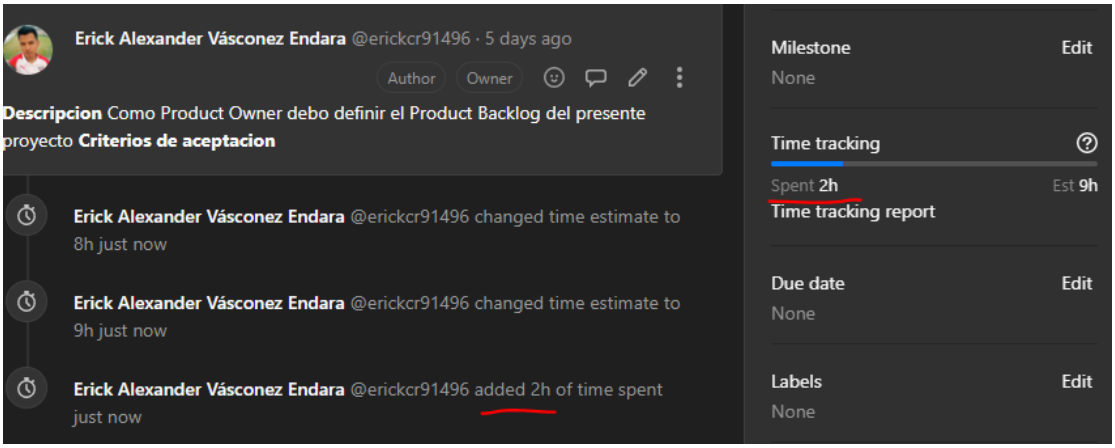


Figura 19 Barra de progreso time tracking

Fuente: Propia

Definir el Weight (Peso, ponderación)

Cuando se tiene muchos Issues, puede ser difícil obtener una visión general. Con los Issues priorizados o puntuados, se puede tener una mejor idea de cuánto tiempo, valor o complejidad tiene o representa un Issue determinado.

1. Establecer el peso o ponderación

En la opción Weight se establece un valor entero no negativo de 0,1,2, etc. Ver Fig.20

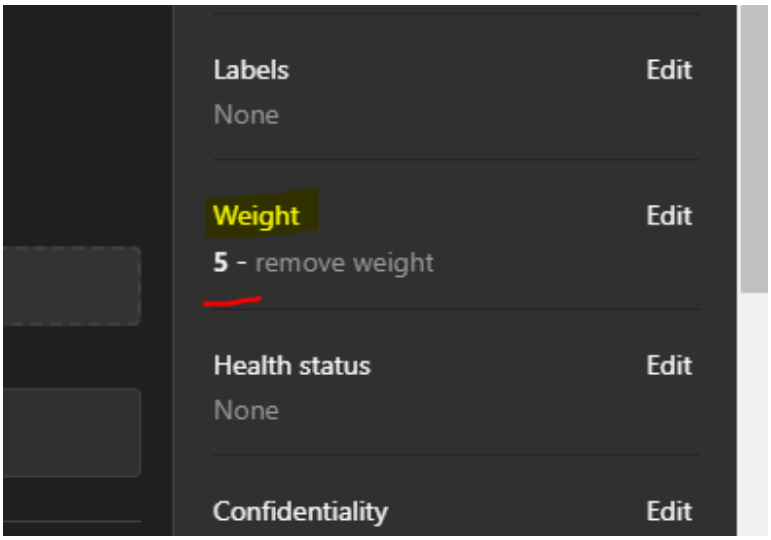


Figura 20 Weight de un issue
Fuente: Propia

2.3.6.2. Fase 2. Juego o Desarrollo

2.1. Sprint Planning

1. Para realizar la reunión del sprint planning se crea un nuevo issue y se etiqueta como reunión. Ver Fig.21

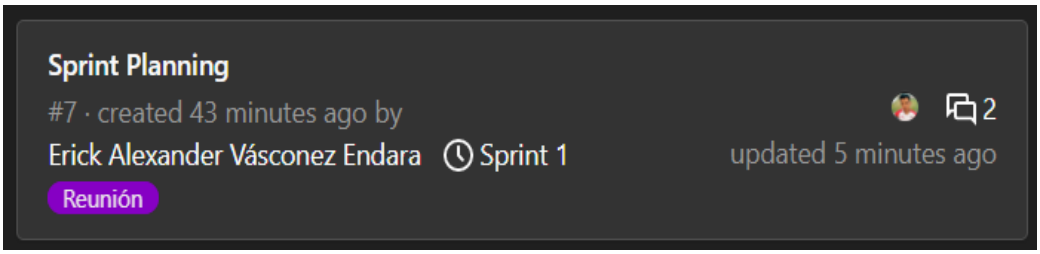


Figura 21 Issue Sprint Planning
Fuente: Propia

2. Dentro del issue se procede a adjuntar texto o archivos que sirvan de apoyo para la reunión. Ver Fig. 22

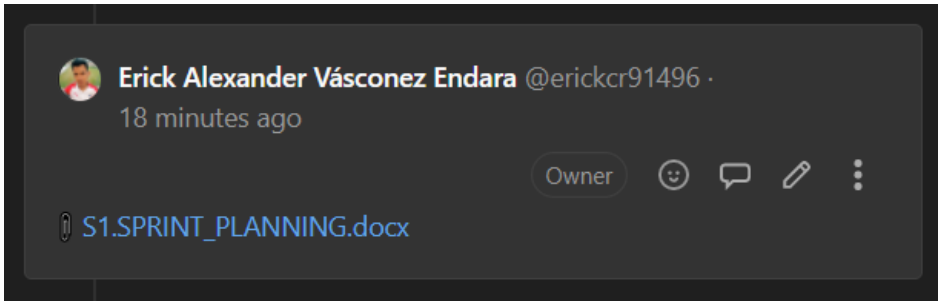


Figura 22 Adjuntos del issue
Fuente: Propia

2. 2. Elaboración del Sprint Backlog

En este punto se debe asignar las historias de usuario definidas en el Product Backlog a cada Sprint correspondiente (Sprint Backlog) especificando la estimación y prioridad de cada una y las fechas de inicio y fin de cada Sprint como se puede ver en el ejemplo de la tabla #

Tabla 7 Backlog del Sprint

ID	Historia de Usuario	Estimación/ Esfuerzo	Sprint	Fecha
Número de orden	Nombre de historia de usuario	Tiempo estimado de duración	Sprint al que pertenece	Fecha inicio y fin del sprint

Fuente: Propia

Para planificar el Sprint Backlog en GitLab se debe crear los Sprint (Milestones) en los cuales definimos el título del Sprint, la descripción del objetivo y la fecha de inicio y fin de cada Milestone.

Asignar milestones

Una vez creados los Milestones procedemos a seleccionar las Issues del Sprint Backlog que serán asignadas a los diferentes Milestones (Sprint 1,..Sprint n) . Ver Fig. 23

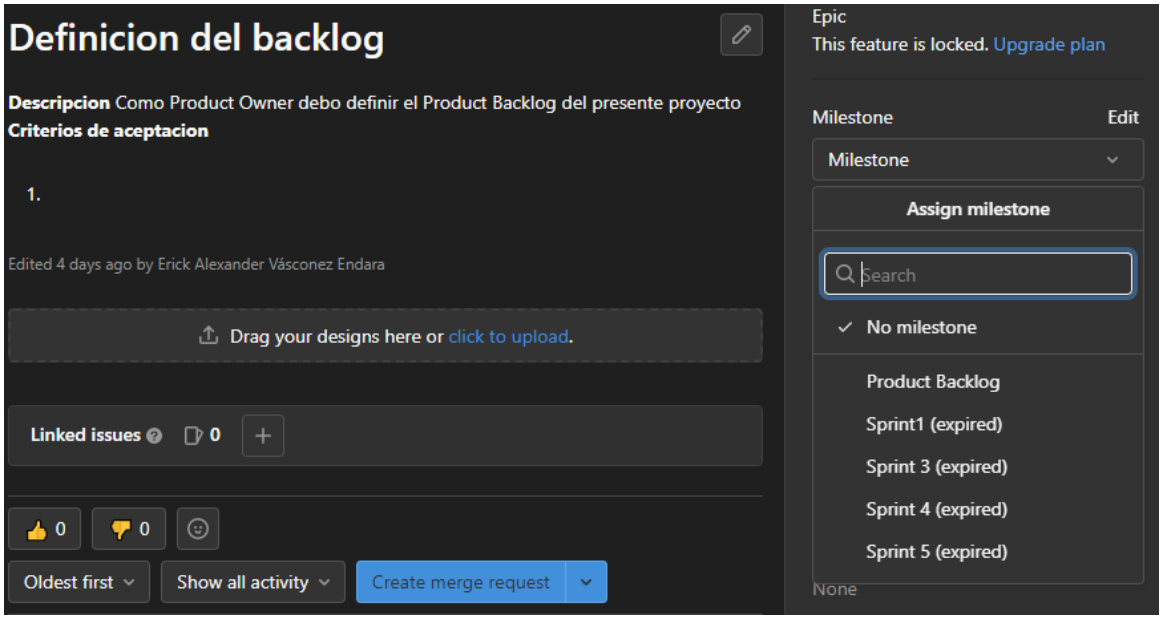


Figura 23 Asignar milestones
Fuente: Propia

2.3 Iteraciones del prototipado

Una vez definidos los roles, historias de usuario, el product backlog se procede a ejecutar las tareas o actividades asignadas del Sprint Backlog.

Iteraciones

- Iteración 1, 2,3, …, N

Se procede a realizar los pasos descritos a continuación por cada una de las iteraciones de los Sprint

Desarrollo del protipado del Sprint

Ejecución del sprint backlog correspondiente. En la Tabla 8 se

- **Sprint #1**
- **Sprint #N**

Tabla 8 Sprint backlog plantilla

Sprint #		
Fecha inicio- Fecha fin		
Historia de Usuario	**Tarea**	**Horas**
Hisotoria Usuario 1	Tarea 1	# horas
	Tarea 2	# horas
Historia de Usuario N	Tarea N	
	Total	# Sumatoria horas

Fuente: Propia

Gestión del prototipo con Boards

Para gestionar el desarrollo del Sprint se hace uso de los Tableros o Boards que es una herramienta de GitLab que permite dar seguimiento a las historias de usuario asignadas al Milestone (Sprint) correspondiente o a lo que llamamos Sprint Backlog y que también permite visualizar y controlar el estado de las tareas asignadas.

En la Fig.24 podemos ver como se organizaron los Sprints en el Board.

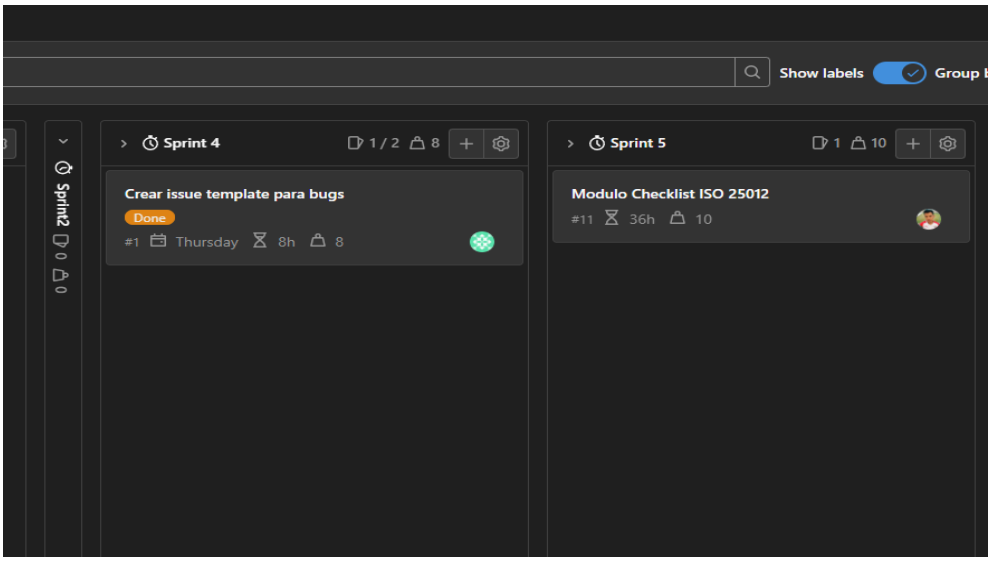


Figura 24 Gestión de boards
Fuente: Propia

1. Para priorizar una tarea o issue se puede crear labels (etiquetas) principales que son: To do (Por hacer), Doing (Ejecutando) y Done (Terminado) colocándolas en columnas dentro del Board. Podemos crear más de un Board para administrar el prototipo para una mejor visualización. Ver Fig.25

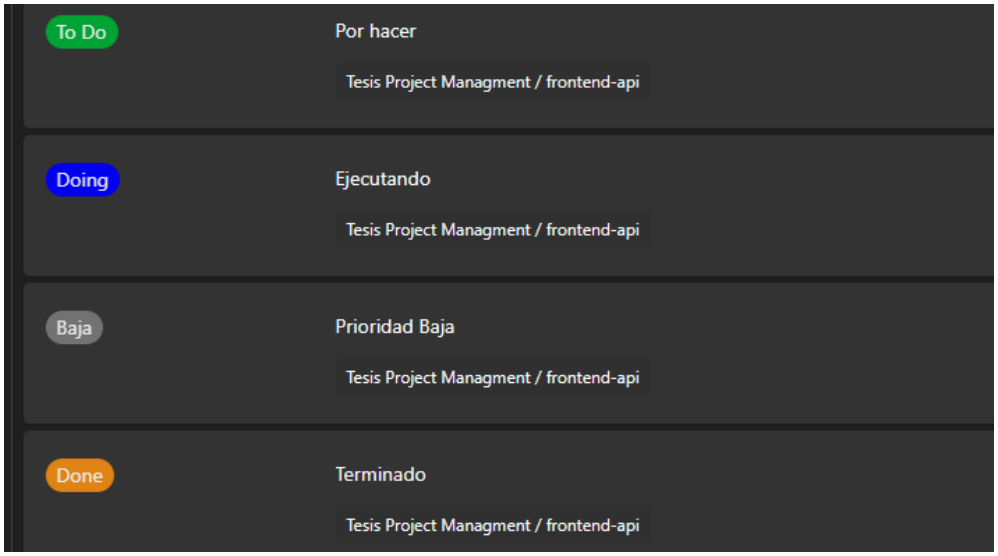


Figura 25 Etiquetas para organizar issues
Fuente: Propia

Cierre del prototipo

- Se procede a etiquetar los Issues terminados trasladándolos a la columna (Done)
- En el **Board** principal, se trasladan los issues terminados del sprint a la columna de **Close** para marcarlo como terminado.
- Finalmente procedemos a realizar el Lanzamiento del prototipo en la Fase 3 y seguir con la siguiente iteración.

2.4 Daily Scrum

Para realizar el Daily Scrum se sigue lo mismo pasos de la sección 2.1 ver Fig. 26

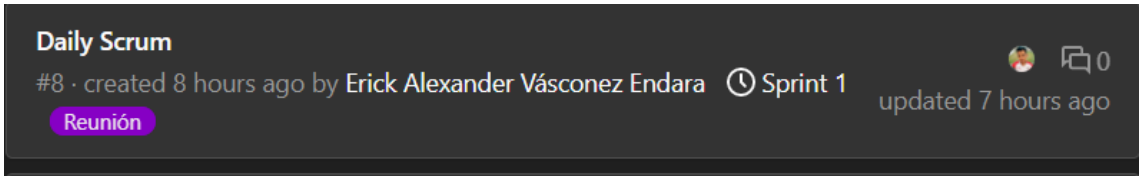


Figura 26 Plantilla issue daily scrum
Fuente: Propia

En la Fig.27 se muestra los adjuntos del issue.

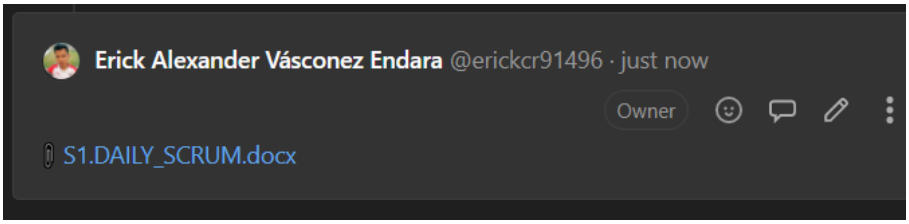


Figura 27 Plantilla adjuntos del issue
Fuente: Propia

2.5 Sprint Review

Para realizarlo se aplican los mismos pasos de la sección 2.1

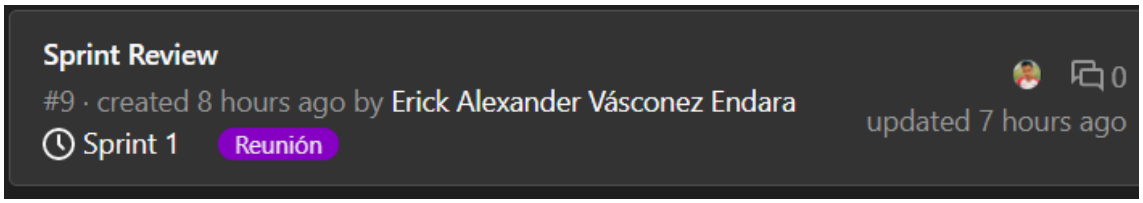


Figura 28 Plantilla sprint review
Fuente: Propia

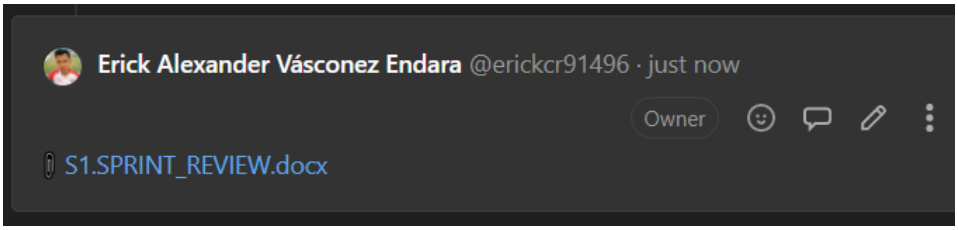


Figura 29 Plantilla recurso sprint review
Fuente: Propia

2.6 Sprint Retrospective

Para realizarlo se aplican los mismos pasos de la sección 2.1

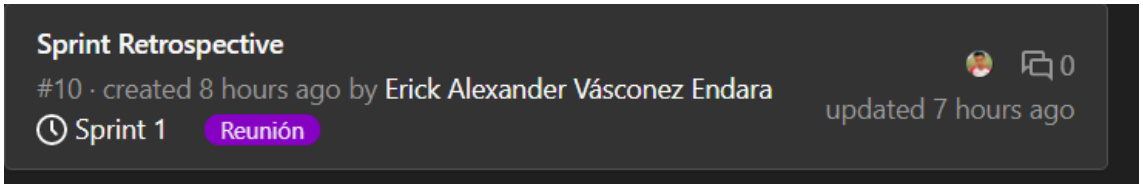


Figura 30 Plantilla sprint retrospecive

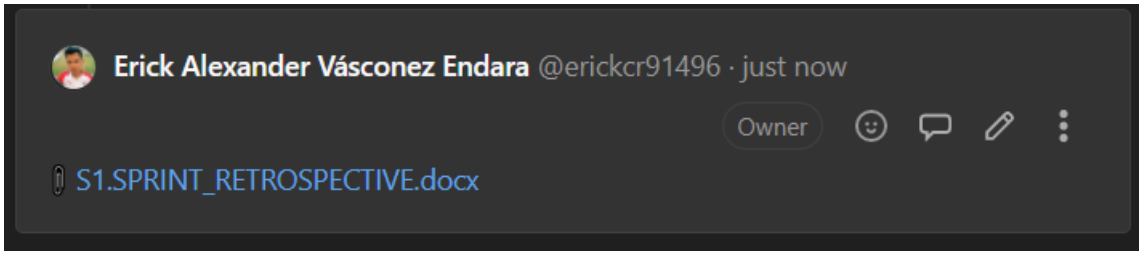


Figura 31 Plantilla recurso sprint retrospective

Fuente: Propia

2.1.6.3. Fase 3. Post-Juego

3.1 Incremento

- Incremento Sprint 1
- Incremento Sprint N

Los incrementos o entregas del producto se realizan mediante Tags y Releases del prototipo funcional.

Etiquetar lanzamiento (Tag)

Los tags se los puede crear desde el modo gráfico en GitLab o desde comandos con Git.

Crear un tag desde GitLab

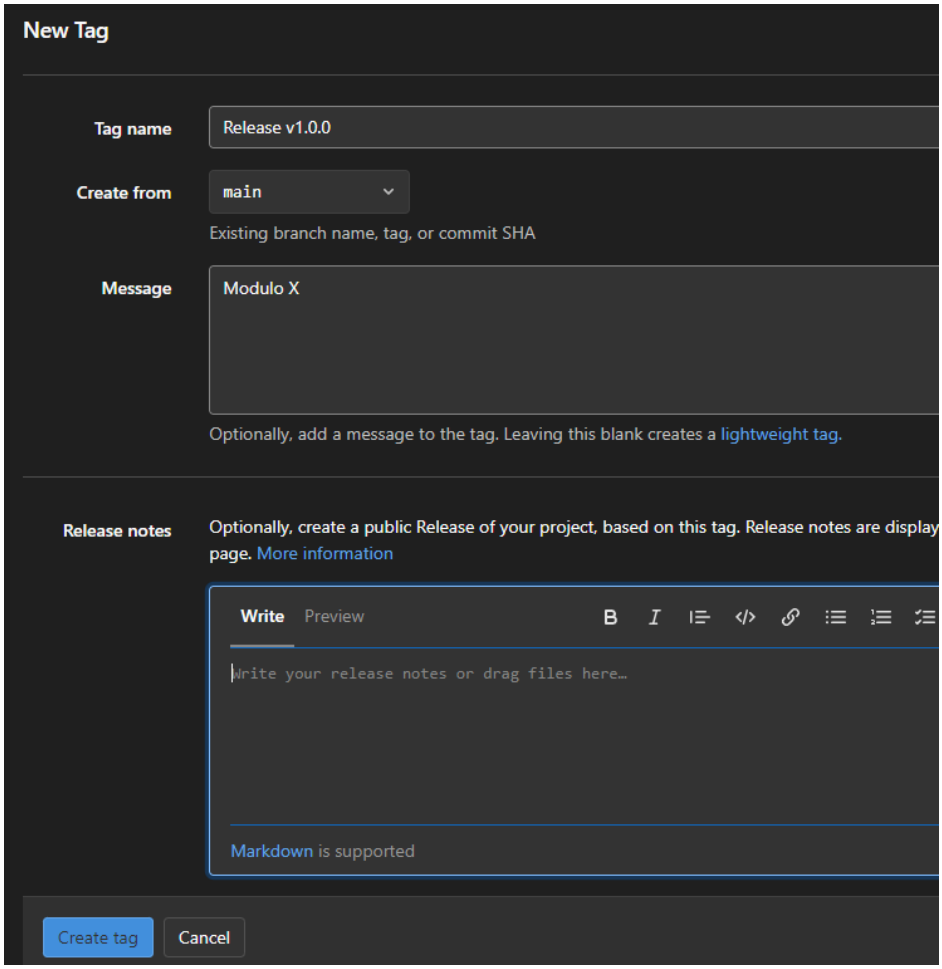


Figura 32 Crear tag en gitlab
Fuente: Propia

Releases

Se basan en tags y son versiones del producto funcional entregados como incremento en cada Sprint. Los releases contienen la identificación, información y los entregables del incremento del sprint

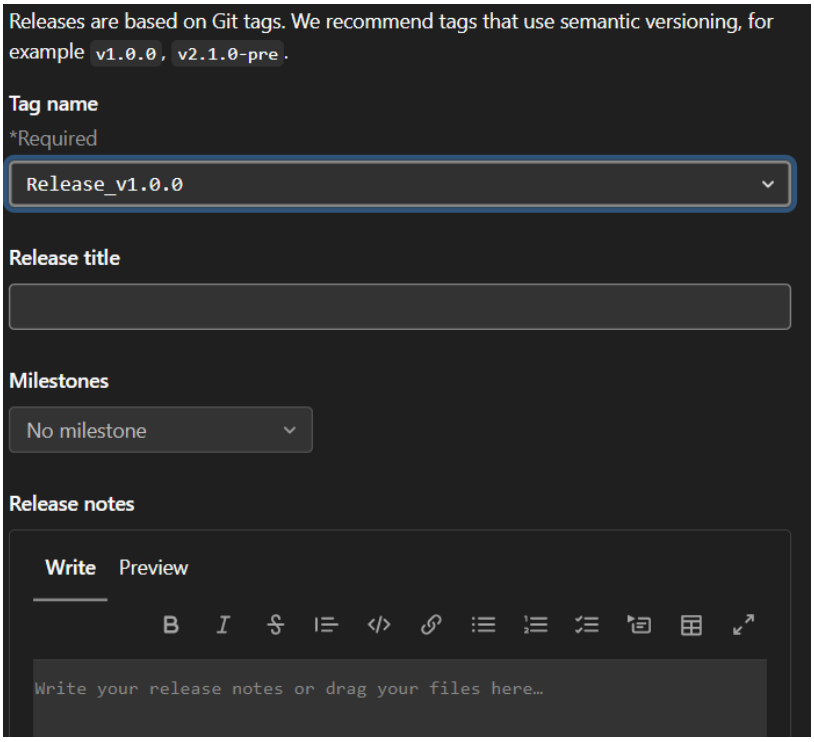


Figura 33 Crear release en gitlab

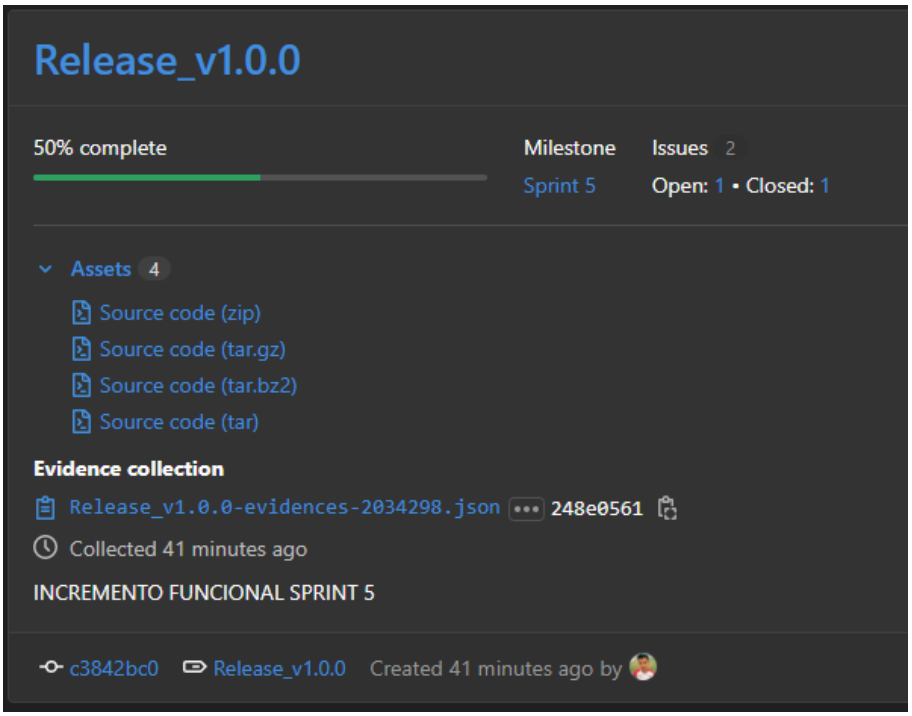


Figura 34 Vista release creado

Fuente: Propia

2.1.7. Formatos – Diagramas de flujo

2.1.7.3. Fase 1. Pre-Juego

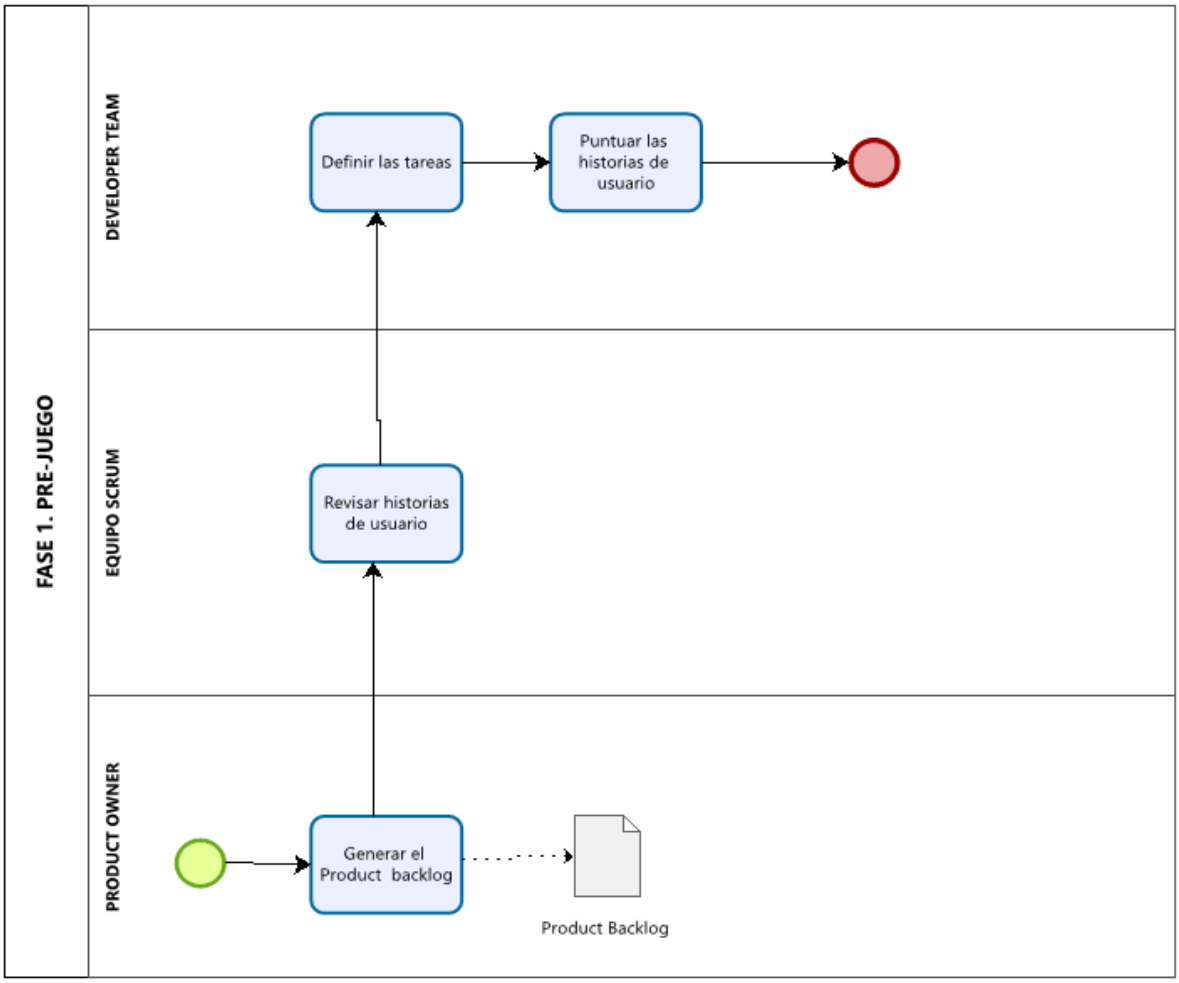


Figura 35 Diagrama fase 1
Fuente: Propia

2.1.7.4. Fase 2. Juego o Desarrollo

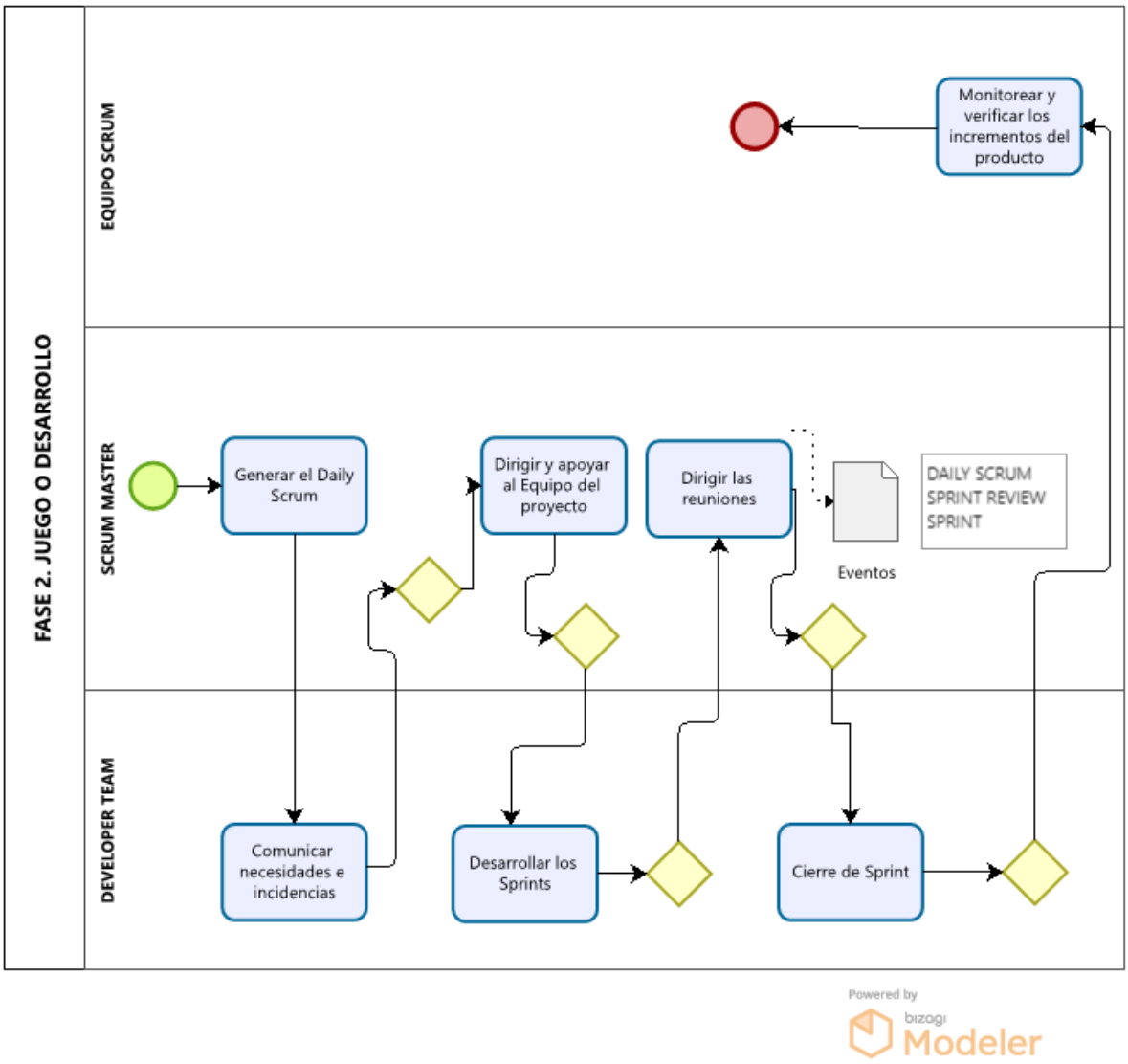


Figura 36 Diagrama fase 2
Fuente: Propia

2.1.7.5. Fase 3. Post-Juego

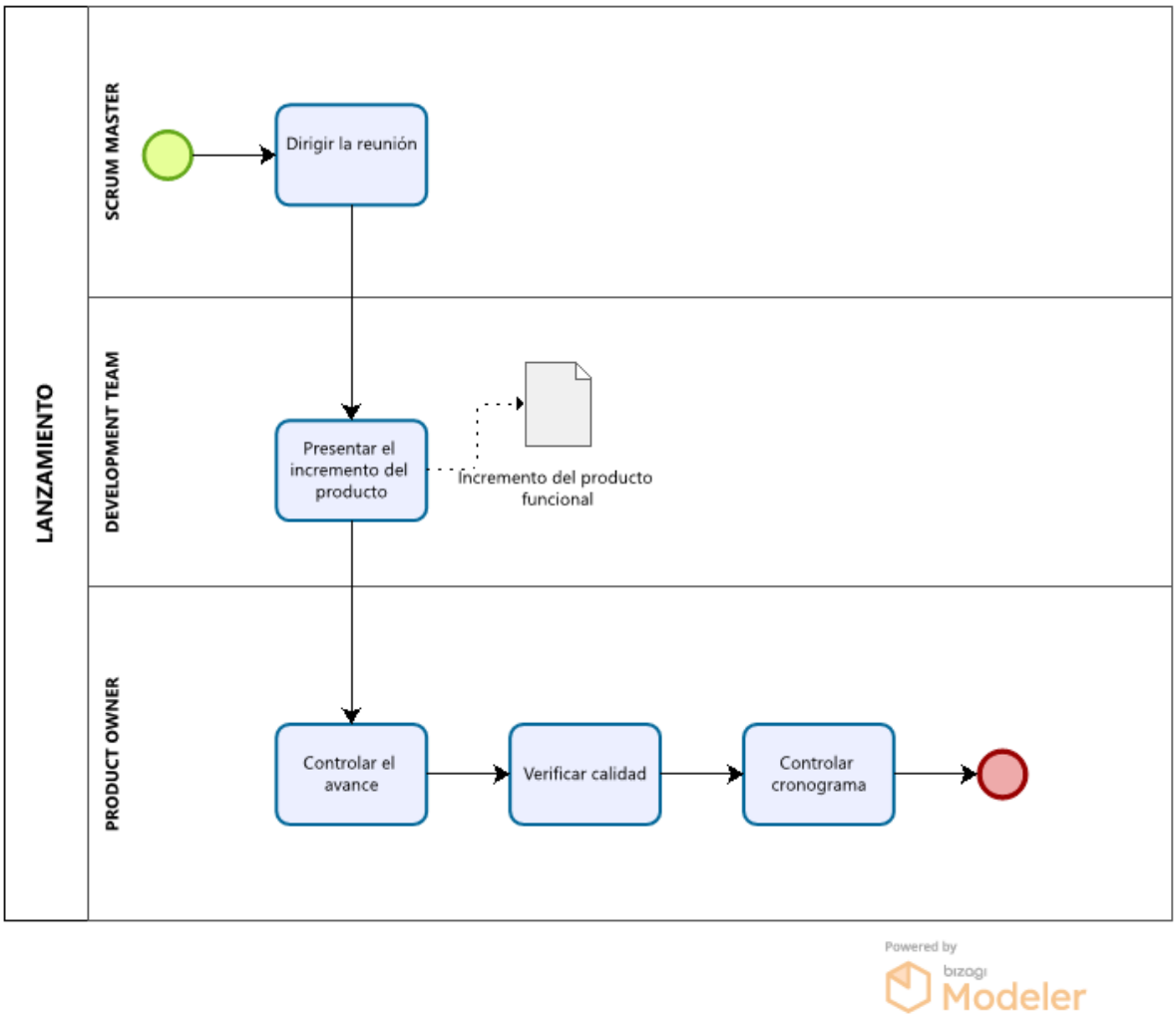


Figura 37 Diagrama fase 3
Fuente: Propia

2.4. Desarrollo de la prueba de concepto

2.4.1. Fase 1. Pre-Juego

1.1 Inicio del proyecto

Creación del proyecto

Creamos nuestro proyecto con el nombre de **app-mevn-validacion-iso.** Para utilizar todos los beneficios gratuitos de GitLab se debe de crear el proyecto en modo Público

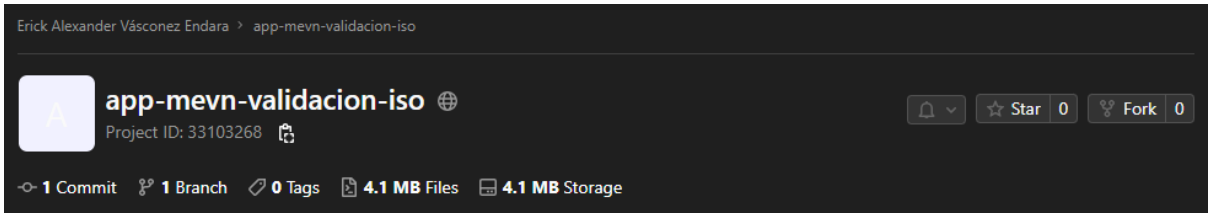


Figura 38 nombre del proyecto creado
Fuente: Propia

Subir código fuente

Git global setup

```
git config --global user.name "Erick Alexander Vásconez Endara"
git config --global user.email "eavasconeze@utn.edu.ec"
```

Create a new repository

```
git clone git@gitlab.com:erickcr91496/testproject.git
cd testproject
git switch -c main
touch README.md
git add README.md
git commit -m "add README"
git push -u origin main
```

Push an existing folder

```
cd existing_folder
git init --initial-branch=main
git remote add origin git@gitlab.com:erickcr91496/testproject.git
git add .
git commit -m "Initial commit"
git push -u origin main
```

Push an existing Git repository

```
cd existing_repo
git remote rename origin old-origin
git remote add origin git@gitlab.com:erickcr91496/testproject.git
git push -u origin --all
git push -u origin --tags
```

Figura 39 Comandos para inicializar repositorio GitLab
Fuente: Propia

Una vez inicializado el respositorio, se cargará la página principal del proyecto

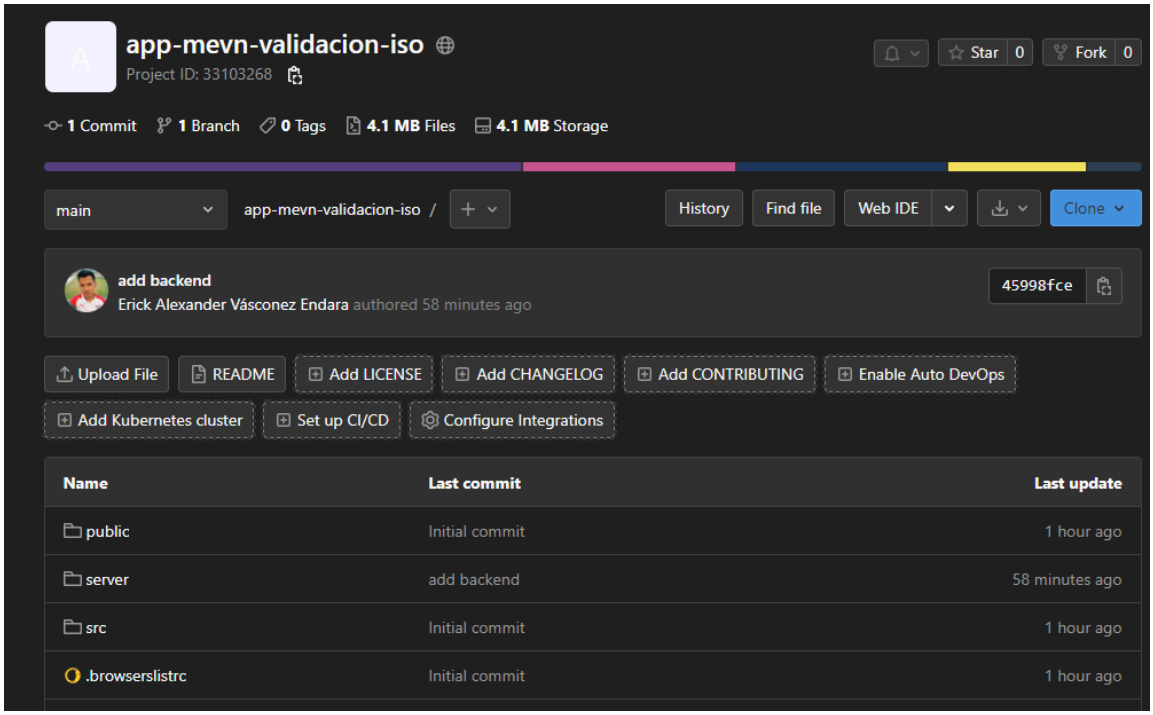


Figura 40 Vista general del repositorio creado
Fuente: Propia

1.2 Definición de roles de scrum

Tabla 9 Roles de scrum proyecto

Rol	Nombre
Product Owner	MSc. Mauricio Rea
Scrum Master	Sr. Erick Vásconez
Development Team	Sr. Erick Vásconez

Fuente: Propia

Agregar los miembros y roles del equipo al proyecto

.

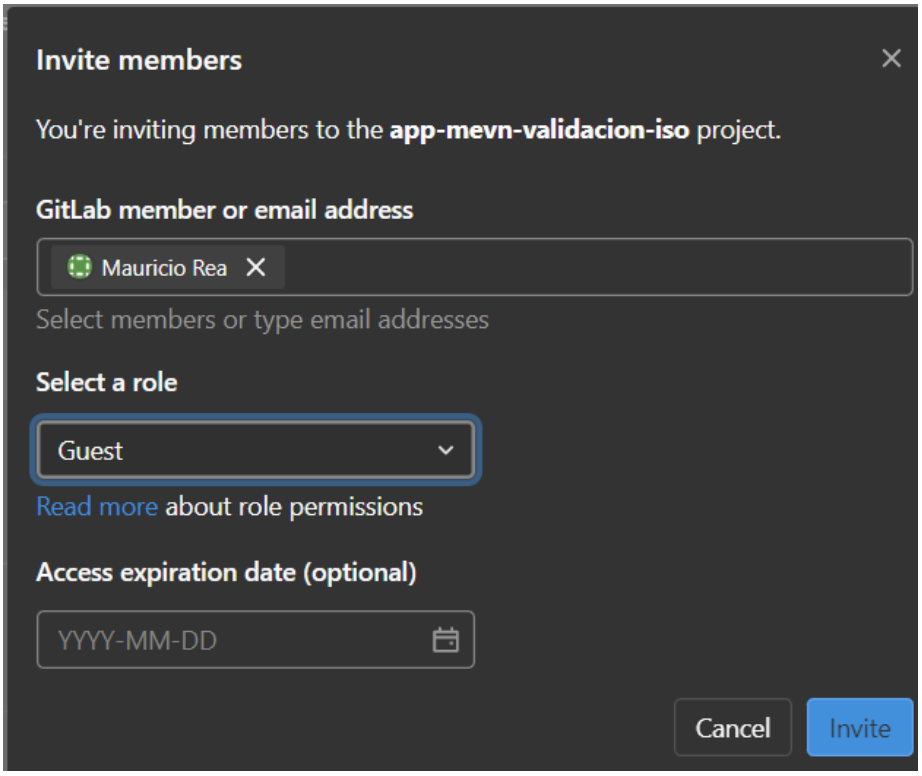


Figura 41 Agregar miembros y roles del equipo

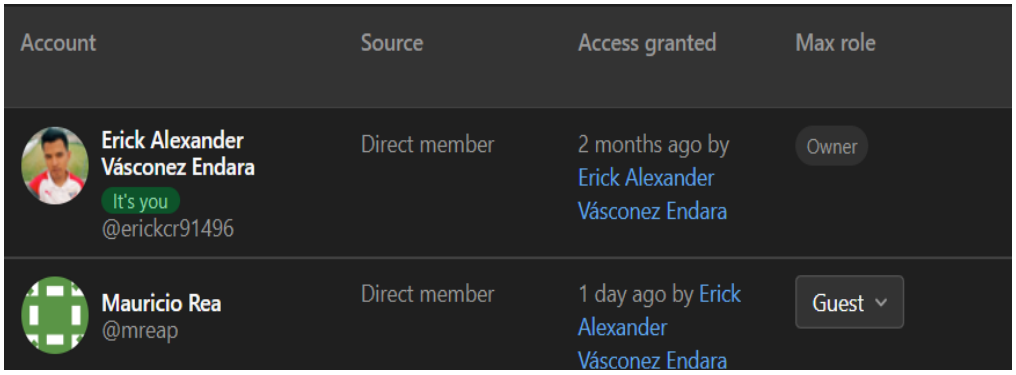


Figura 42 Vista de los miembros agregados

Fuente: Propia

1.3 Definición del Product Backlog

Tabla 10 Product backlog del proyecto

ID	Historias de Usuario	Prioridad	Estimación	Descripción

1	Login/Registro de Ususarios (Backend)	BAJA	19 horas	Creación de la base de datos con las tabla de usuarios y funcionalidad de Login y Registro
2	Creación de proyectos a evaluar	MEDIA	6 horas	Creación de la tabla proyectos, diseño de la vista y crud de proyectos
3	Creación del módulo de características y sub-características de la ISO 25012	ALTA	50 horas	Creación del módulo de checklist de la norma y evaluación de cumplimiento
4	Creación de reporte de evaluación	ALTA	15 horas	Diseño y creación de reporte detallado del proyecto evaluado

Fuente: Propia

Creación de Issues del producto backlog

Se crea los Issues por cada historia de usuario especificando la prioridad (Labels) y la estimación de tiempo.

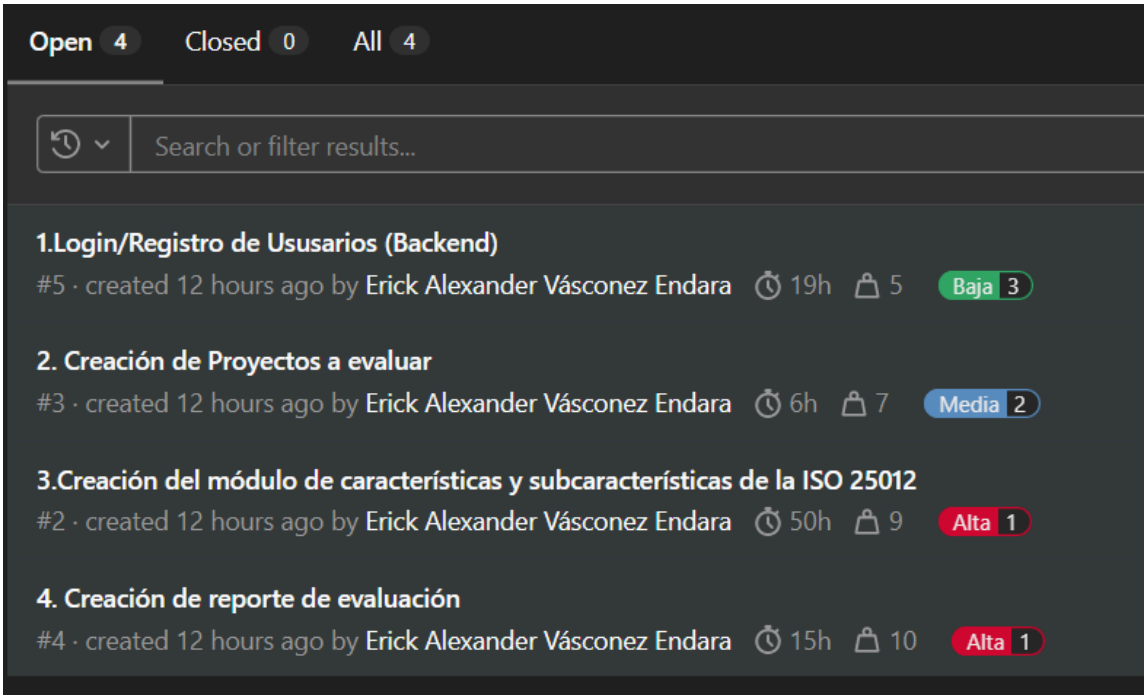


Figura 43 Issues del product backlog en gitlab

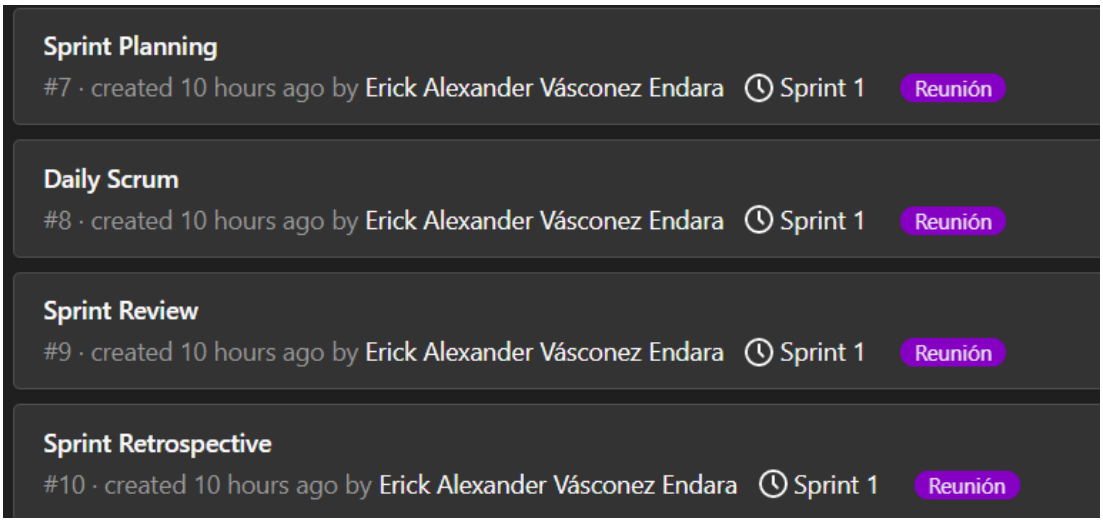


Figura 44 Issues de reuniones de scrum

Fuente: Propia

1.4 Creación de Milestones o Sprints

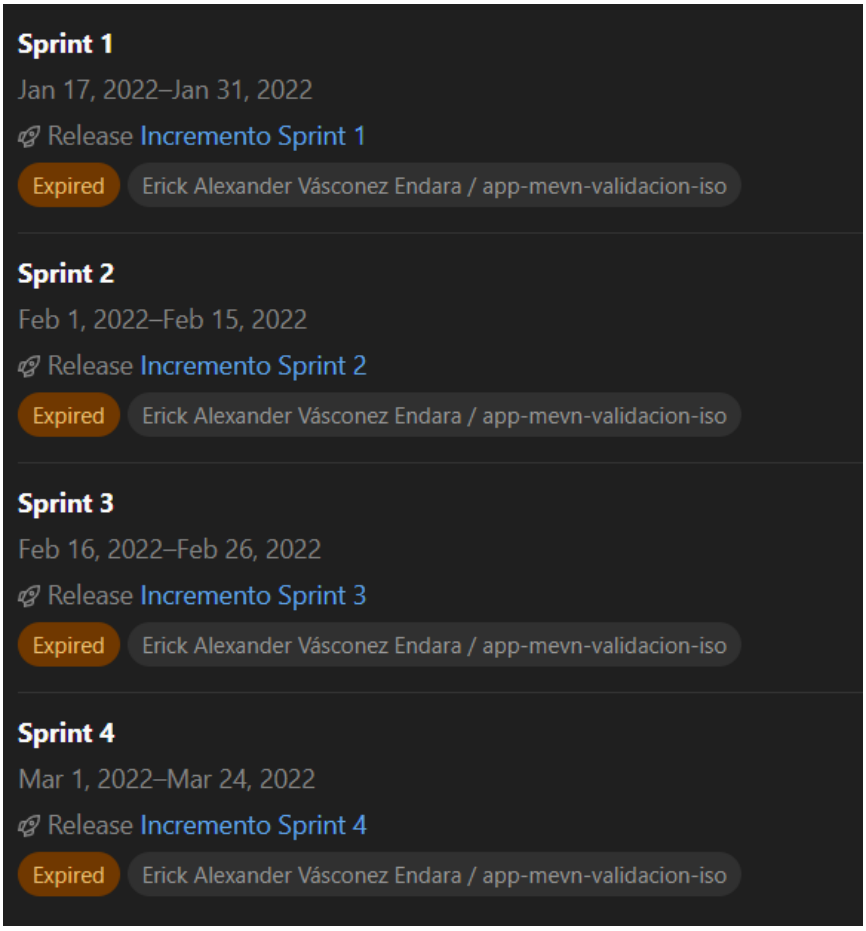


Figura 45 Milestones creados del proyecto
Fuente: Propia

1.5 Definición de tareas de Historias de Usuario

Tabla 11 Historias de usuario 1

HISTORIA DE USUARIO	
ID: 1	**Usuario:** Usuario
Nombre de historia: Login / Registro de Usuarios	
Prioridad: **BAJA (5)**	**Estimación: 19h**
Descripción:	
Diseño y construcción del módulo de inicio de sesión (Login) y registro de usuarios. Las tareas a realizar son: 1. Creación, instalación y configuración del Proyecto	

2. Diseño de la vista de Login y Registro
3. Creación de la tabla users en la base de datos
4. Desarrollo de la vista de registro
5. Creación del método de registro y login de usuarios

Observaciones:

Fuente: Propia

Tabla 12 Historia de usuario 2

HISTORIA DE USUARIO

ID: 2 **Usuario:** Usuario

Nombre de historia: Creación de proyectos a evaluar

Prioridad: MEDIA (7) **Estimación: 6h**

Descripción:

Diseño y construcción del módulo de registro de proyectos. Las tareas a realizar son:

6. Diseño de la vista de de proyectos
7. Creación de la tabla proyectos en la base de datos
8. Creación del CRUD de proyectos

Observaciones:

Fuente: Propia

Tabla 13 Historia de usuario 3

HISTORIA DE USUARIO

ID: 3 **Usuario:** Usuario

Nombre de historia: Módulo de Características y Sub-características ISO 27012

Prioridad: **ALTA (10)** **Estimación: 50h**

Descripción:

Diseño y construcción del módulo de características y subcaracterísticas de la norma ISO 27012. Las tareas a realizar son:

9. Diseño de la vista de formulario de checklist
10. Creación de la tabla evaluaciones relacionada con proyectos
11. CRUD de la tabla evaluaciones

Observaciones:

Fuente: Propia

Tabla 14 Historia de usuario 4

HISTORIA DE USUARIO

ID: 4 **Usuario:** Usuario

Nombre de historia: Creación de reporte de evaluación

Prioridad: **ALTA (10)** **Estimación: 15h**

Descripción:

. Las tareas a realizar son:

12. Diseño del reporte de evaluación detallado
13. Descargar reporte en pdf

Observaciones:

Fuente: Propia

Lista de tareas de issue

Definimos las tareas o actividades a realizar dentro de cada Issue del producto backlog mediante listado de tareas

- **Issue #1**

  Listado de tareas de historia 1

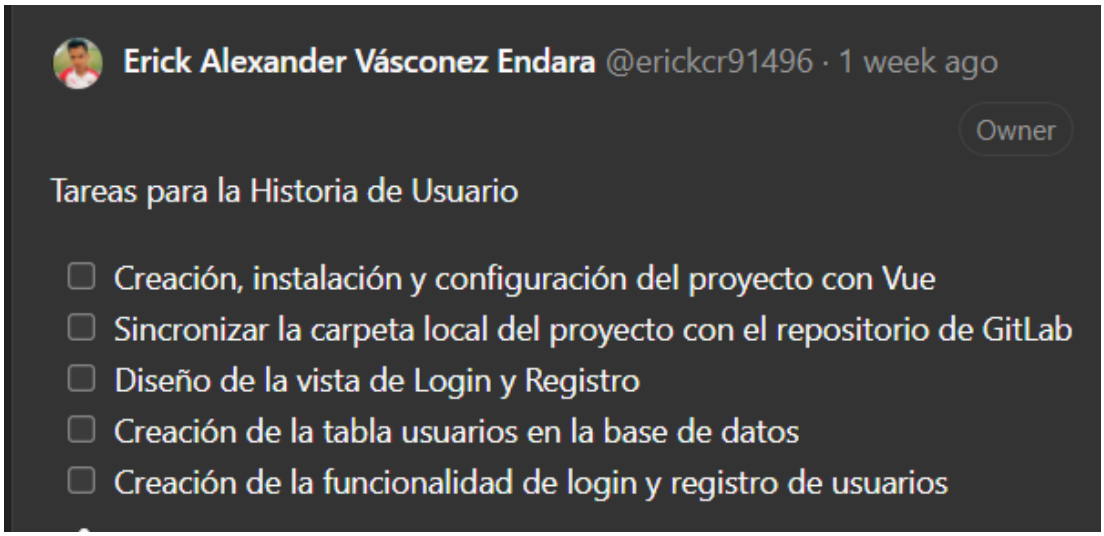


Figura 46 Tareas del issue 1
Fuente: Propia

- **Issue #2**

  Listado de tareas de historia 2

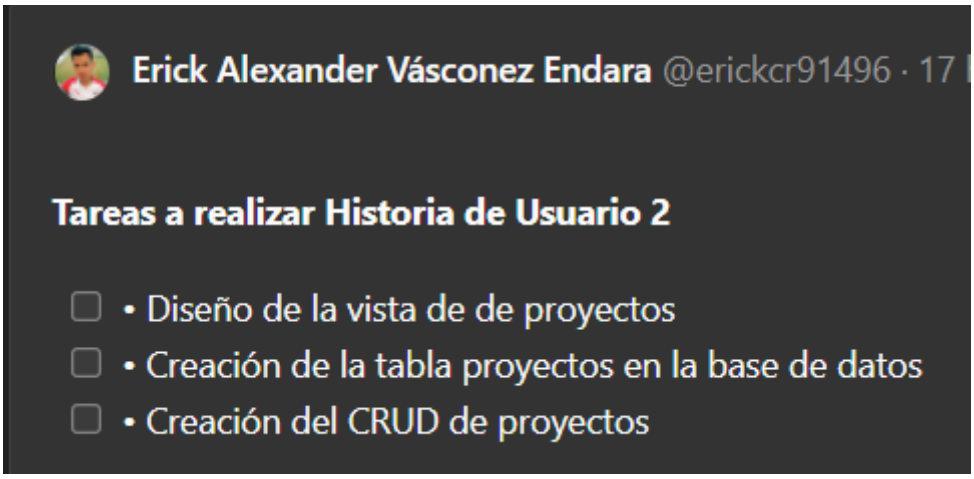


Figura 47 Tareas del issue 2
Fuente: Propia

- **Issue #3**

  Listado de tareas de historia 3

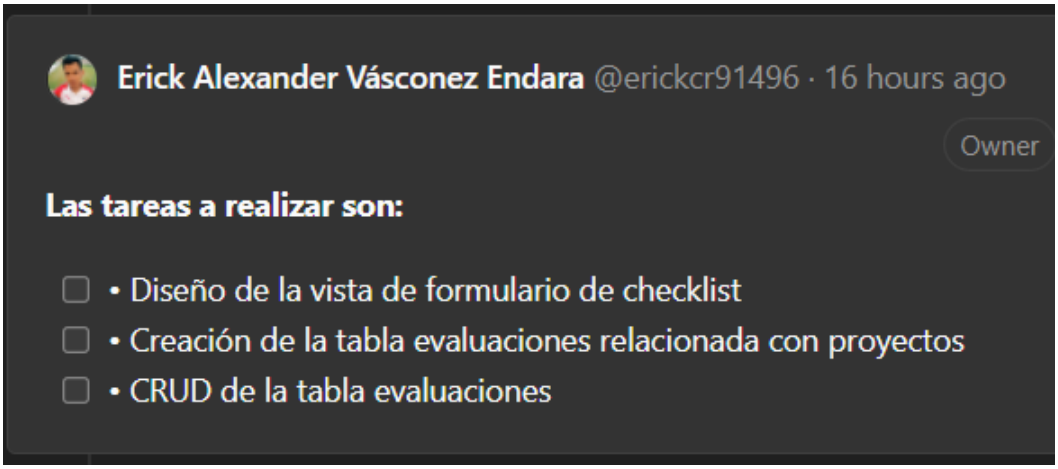


Figura 48 Tareas del issue 3
Fuente: Propia

- **Issue #4**

  Listado de tareas de historia 4

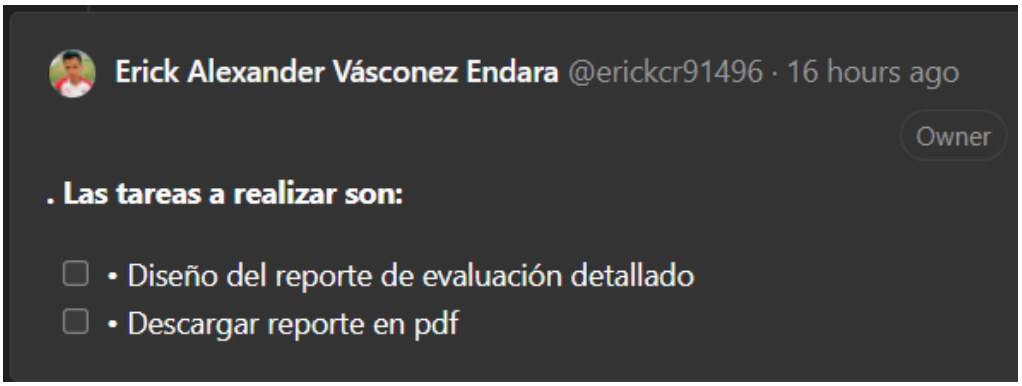


Figura 49 Tareas del issue 4
Fuente: Propia

Time tracking del Issue

- **Issue #1**

  Estimación del Issue #1 de 19 horas

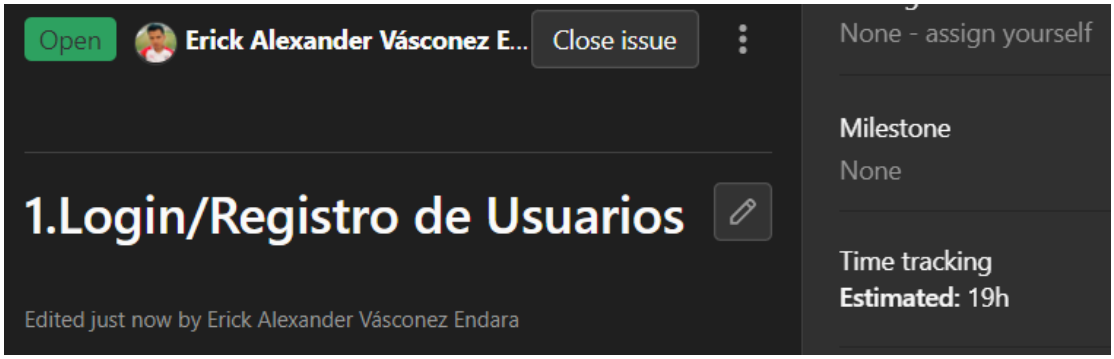


Figura 50 Time tracking issue 1
Fuente: Propia

- **Issue #2**

Estimación del Issue #2 de 6 horas

Figura 51 Time tracking issue 2
Fuente: Propia

- **Issue #3**

Estimación del Issue #3 de 50 horas

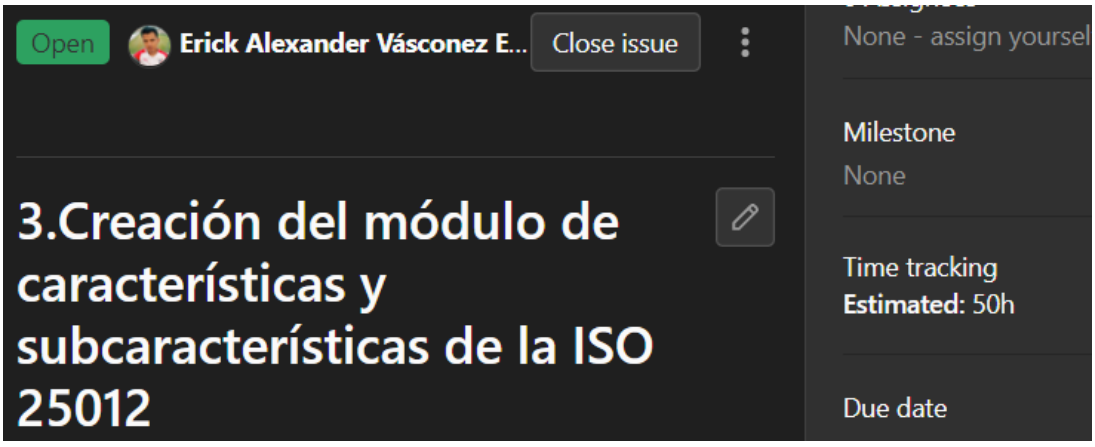


Figura 52 Time tracking del issue 3
Fuente: Propia

- **Issue #4**

Estimación del Issue #4 de 15 horas

Figura 53 Time tracking del issue 4
Fuente: Propia

Definir el Weight (Peso, ponderación)

- **Issue #1**

Prioridad baja, weight 5

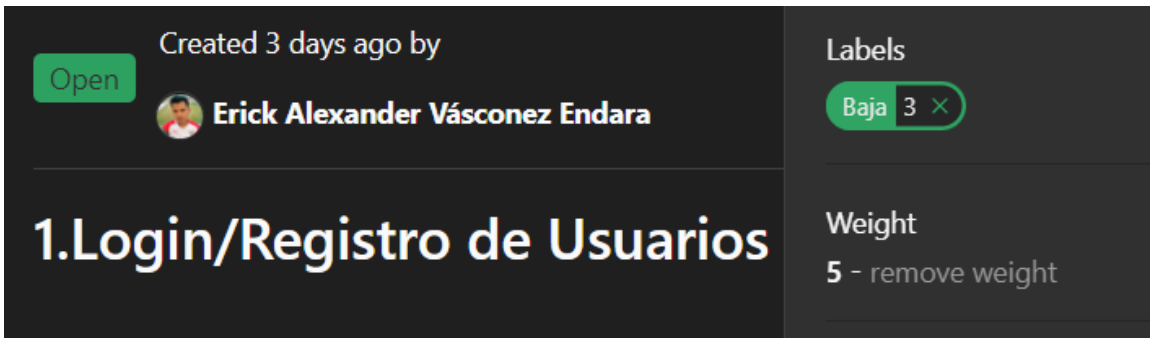


Figura 54 Weight issue 1
Fuente: Propia

- **Issue #2**

Prioridad media, weight 7

Figura 55 Weight issue 2
Fuente: Propia

- **Issue #3**

  Prioridad alta, weight 10

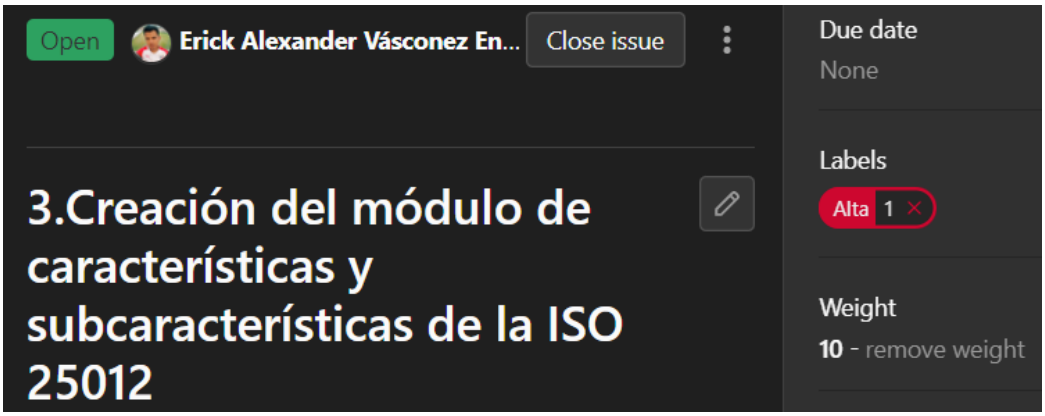


Figura 56 Weight issue 3
Fuente: Propia

- **Issue #4**

  Prioridad alta, weight 10

Figura 57 Weight issue 4
Fuente: Propia

2.4.2. Fase 2. Juego o Desarrollo

2.1. Sprint Planning

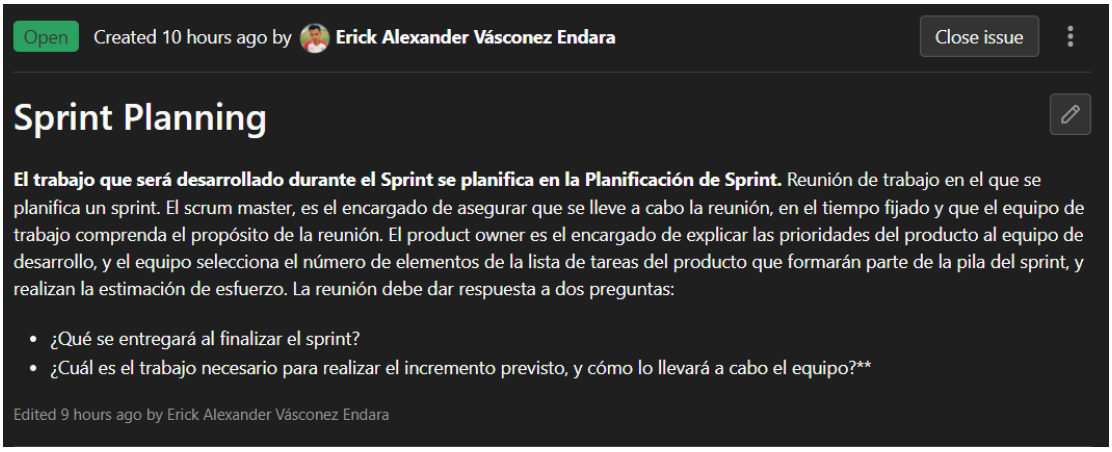


Figura 58 Sprint planning del proyecto

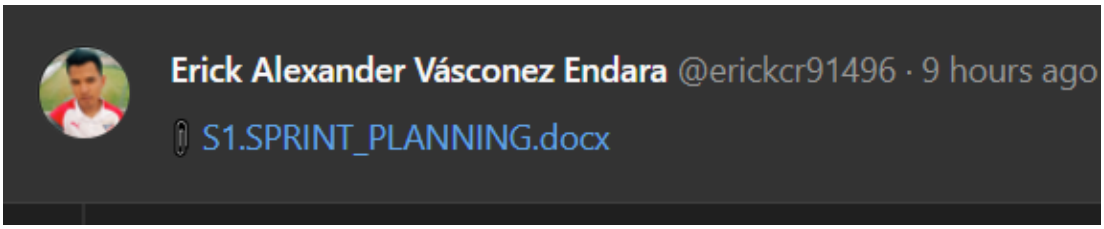


Figura 59 Recursos del sprint planning proyecto

Fuente: Propia

2.2 Sprint backlog

Tabla 15 Sprint backlog del proyecto

ID	Historias de Usuarios	Estimación/ Esfuerzo	Sprint	Fecha
1	Login / Registro de Usuarios	19h	Sprint 1	(17/01/2022) al (31/01/2022)
2	Creación de proyectos a evaluar	6h	Sprint 2	(01/02/2022) al (15/02/2022)
3	Creación del módulo de	50h	Sprint 3	(16/02/2022) al (26/02/2022)

	características y subcaracterísticas de la ISO 25012				
4	Creación de reporte de evaluación	15 h	Sprint 4	(01/03/2022) (24/03/2022)	al

Fuente: Propia

Asignar milestones a issues

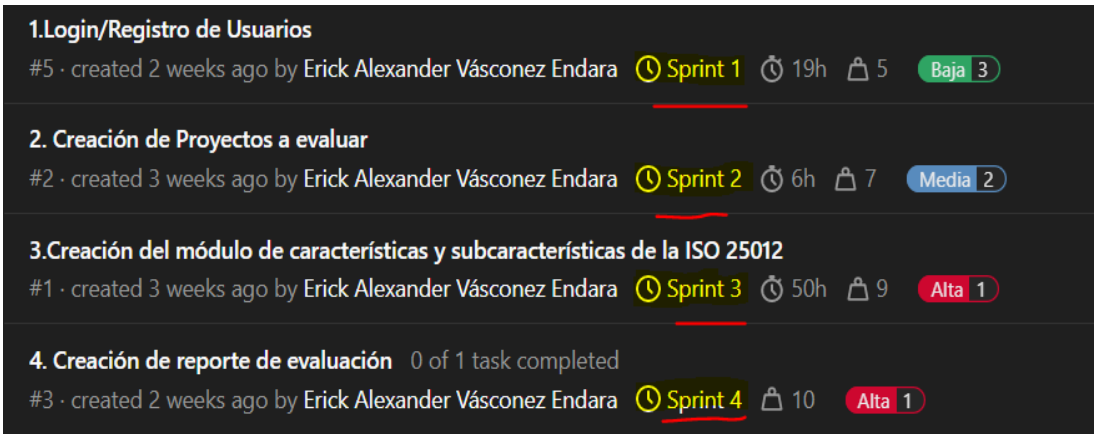


Figura 60 Asignación de milestones a issues del proyecto

Figura 61 Reuniones de scrum del proyecto

Fuente: Propia

2.3 Iteraciones del prototipado

Iteración 1

Desarrollo del prototipo del sprint

Sprint 1

Tabla 16 Prototipo sprint 1

Sprint 1 **17-01-2022- 31-01-2022**		
Historia de Usuario	**Tarea**	**Horas**
Login/Registro de Usuarios	Creación, instalación y configuración del proyecto con Vue	1 h
	Sincronizar la carpeta local del proyecto con el repositorio de GitLab	1h
	Diseño de la vista de Login y Registro	4h
	Creación de la tabla usuarios en la base de datos	2h
	Creación de la funcionalidad de login y registro de usuarios	10h
	Total	18h

Fuente: Propia

Gestión del prototipo con Boards

Sprint 1

- Issues del Sprint 1 (Milestone 1)

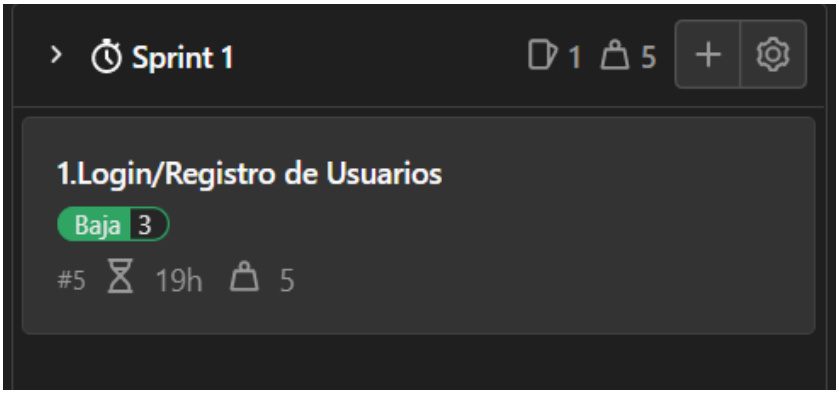


Figura 62 Board sprint 1 issue 1
Fuente: Propia

- Issues Por Hacer (To-Do)

Figura 63 Issues to-do sprint 1

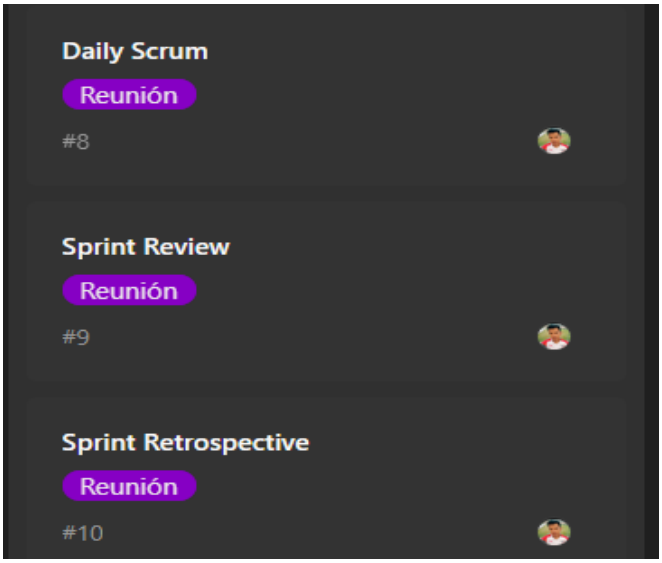


Figura 64 Reuniones board sprint 1

Fuente: Propia

- Issues En Proceso (Doing)

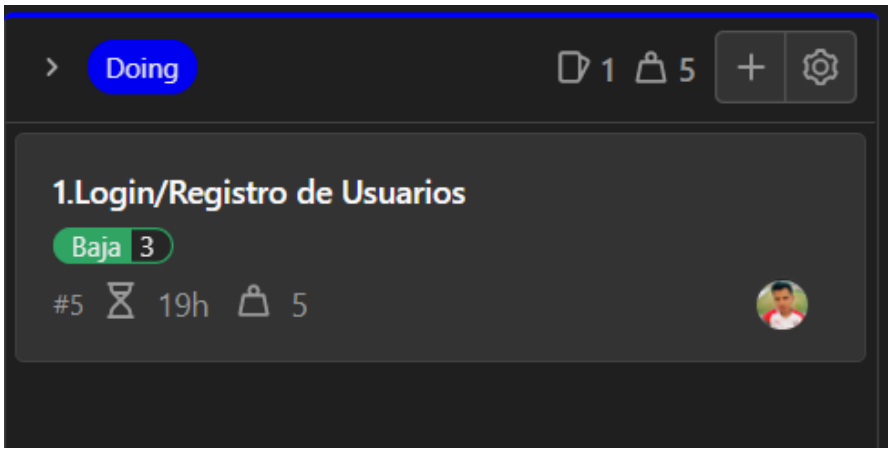


Figura 65 Issue do-ing sprint 1
Fuente: Propia

Cierre del prototipo

- Issues Terminadas (Done)

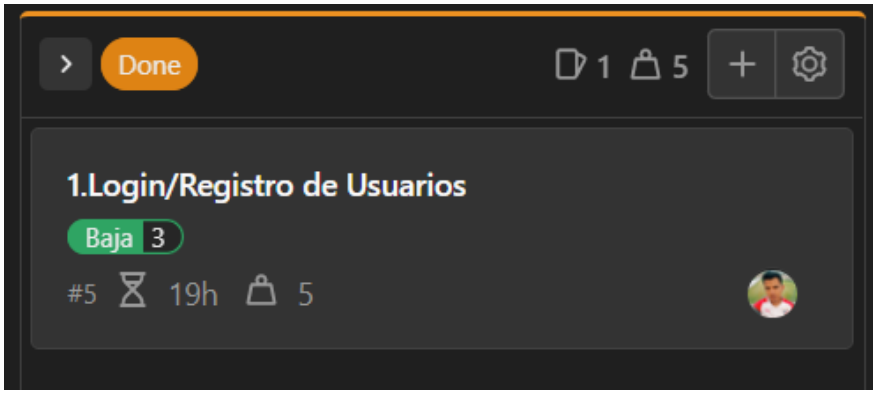


Figura 66 Issue terminada done sprint 1
Fuente: Propia

- Issue cerrado en la columna close del Board

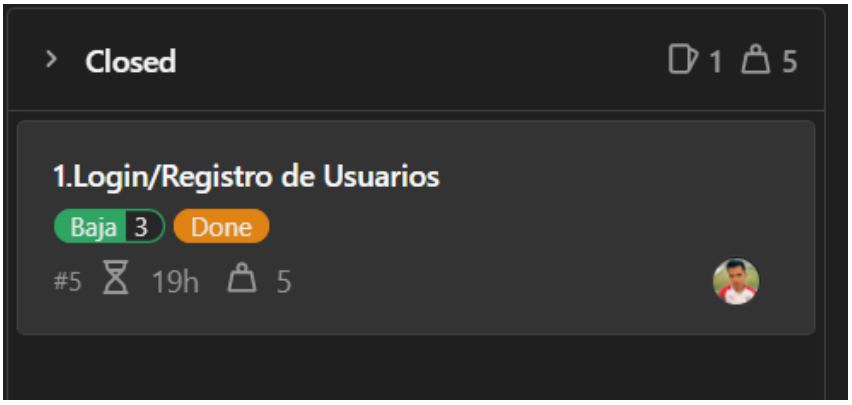


Figura 67 Issue1 cerrado en board sprint 1
Fuente: Propia

Daily Scrum

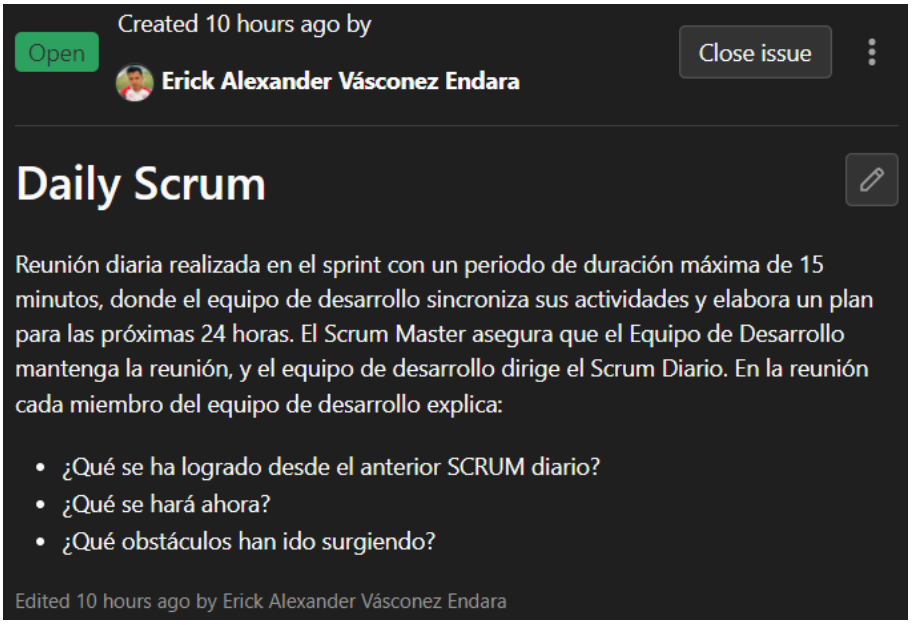


Figura 68 Daily scrum sprint 1

Figura 69 Recurso daily scrum sprint 1

Fuente: Propia

Sprint Review

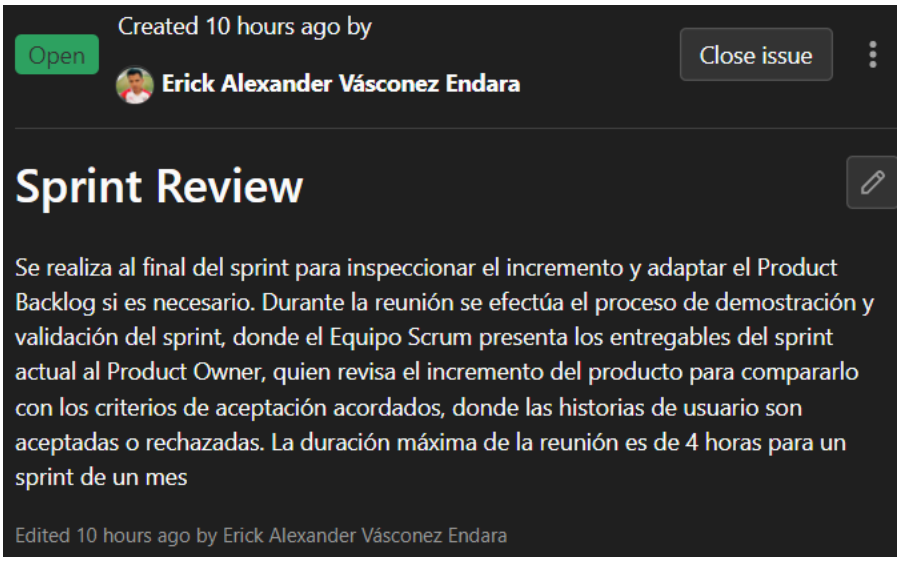


Figura 70 Sprint review milestone 1

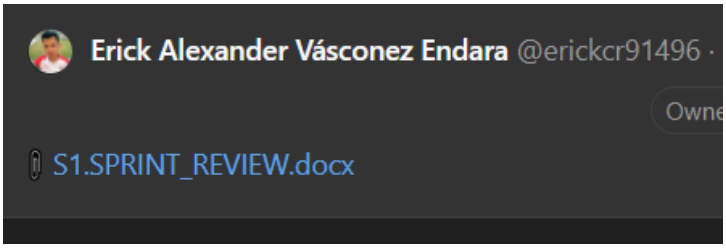


Figura 71 Recurso sprint review milestone 1

Fuente: Propia

Sprint Retrospective

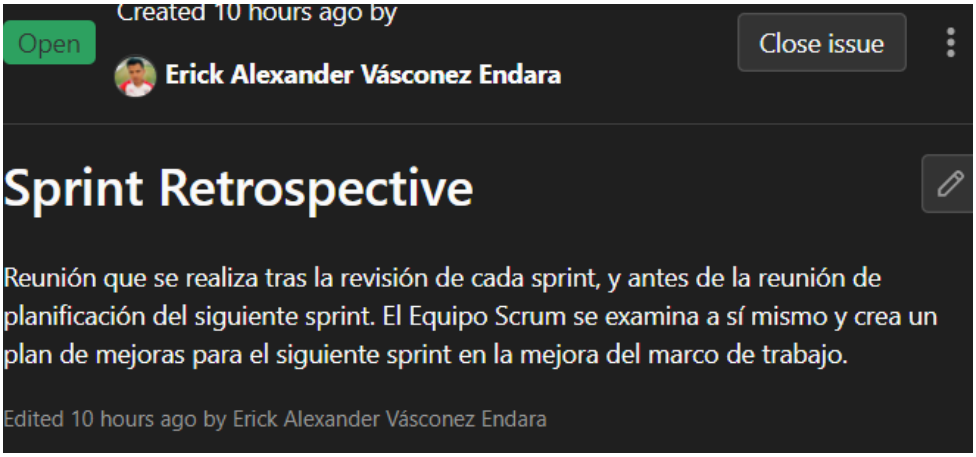


Figura 72 Sprint retrospective milestone 1

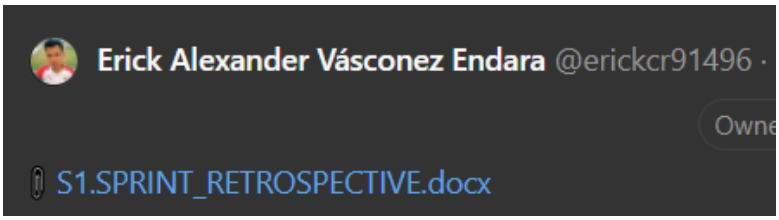


Figura 73 Recurso sprint retrospective milestone 1

Fuente: Propia

Iteración 2

Desarrollo del prototipado del Sprint

Sprint 2

Tabla 17 Prototipo Sprint 2

Sprint 2 01/02/2022- 15/02/2022		
Historia de Usuario	**Tarea**	**Horas**
Creación de Proyectos a evaluar	Diseño de la vista de de proyectos	1h
	Creación de la tabla proyectos en la base de datos	2h
	Creación del CRUD de proyectos	3h
	Total	6h

Fuente: Propia

Gestión del proyecto con Boards

Sprint 2

- Issues del Sprint 2 (Milestone 2)

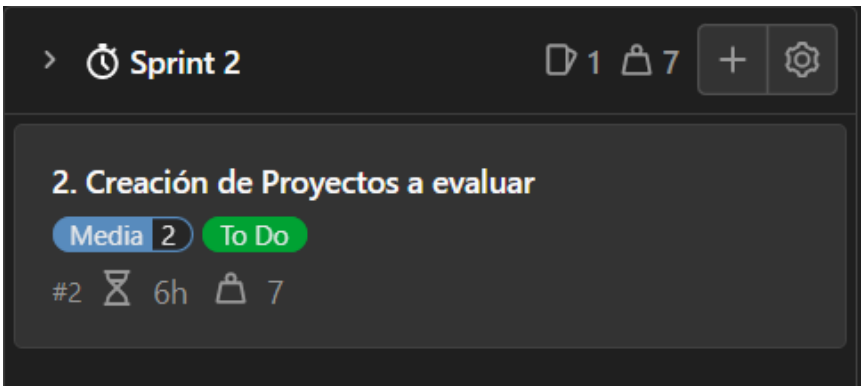


Figura 74 Board issue 2 milestone 2

- Issues Por Hacer (To-Do)

Figura 75 Issues to-do milestone 2

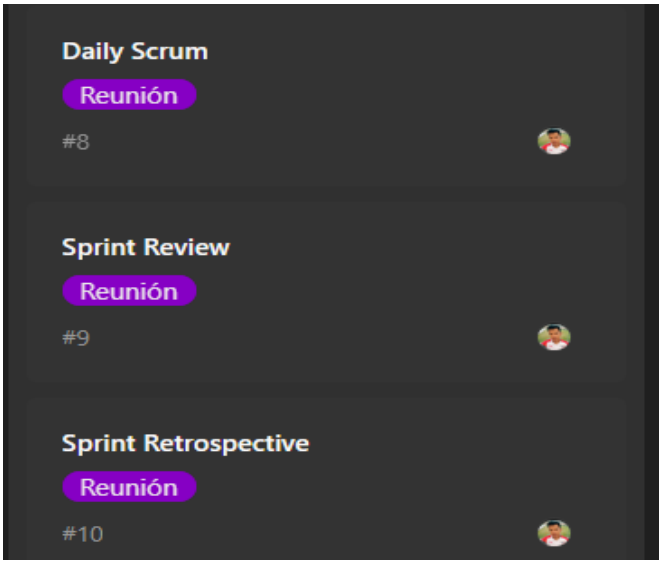


Figura 76 Reuniones milestone 2

Fuente: Propia

- Issues En Proceso (Doing)

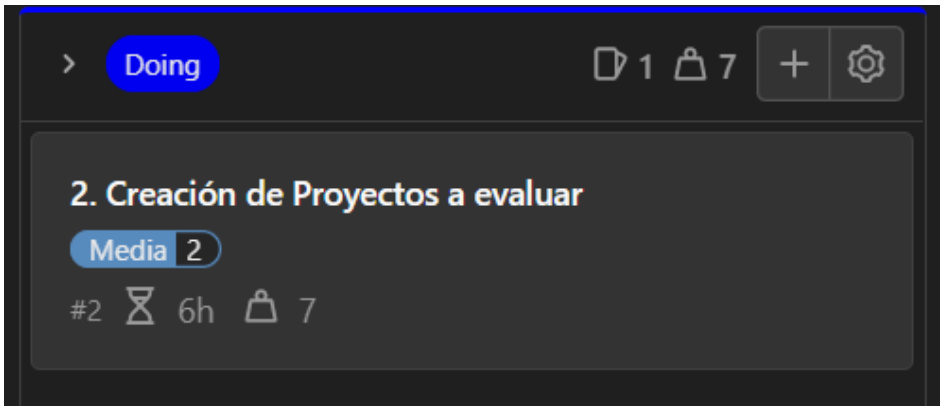


Figura 77 Issue 2 do-ing milestone 2
Fuente: Propia

Cierre del prototipo

- Issues Terminadas (Done)

Figura 78 Issue 2 terminada done
Fuente: Propia

- Issue cerrado en la columna Closed del Board

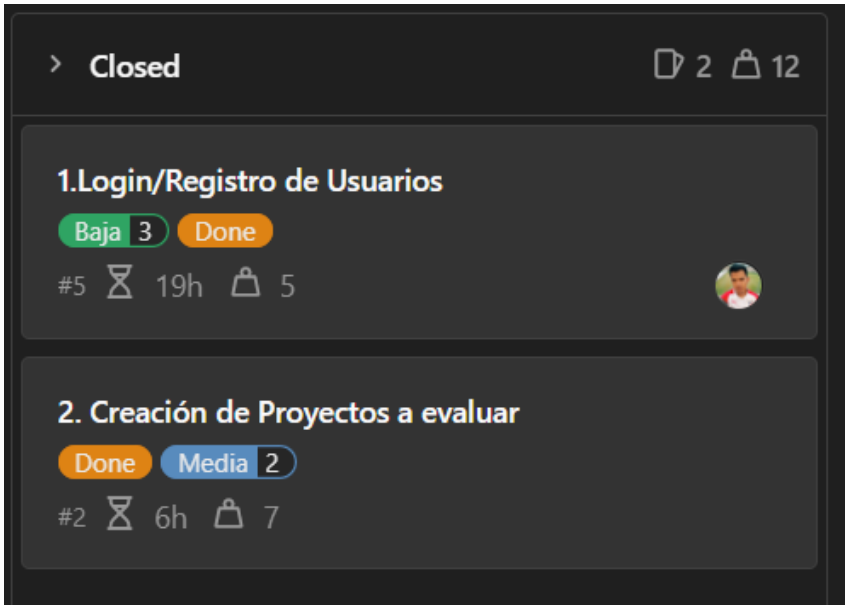


Figura 79 Board issue 2 close milestone 2
Fuente: Propia

Daily Scrum

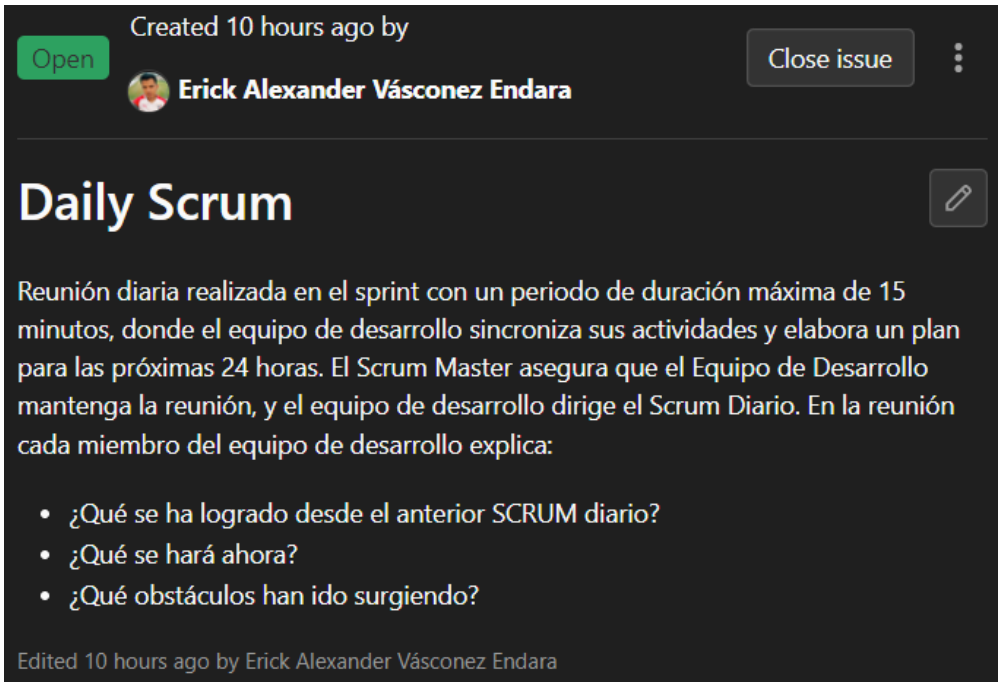


Figura 80 Daily scrum milestone 2

Figura 81 Recurso daily scrum milestone 2

Fuente: Propia

Sprint Review

Figura 82 Sprint review milestone 2

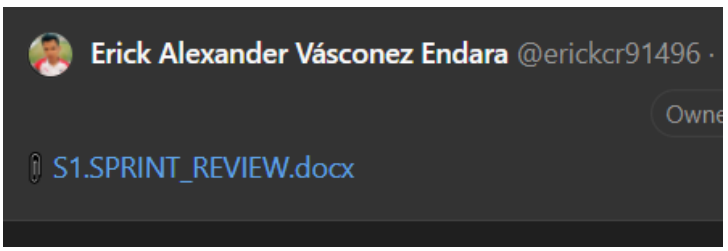


Figura 83 Recurso sprint review milestone 2

Fuente: Propia

Sprint Retrospective

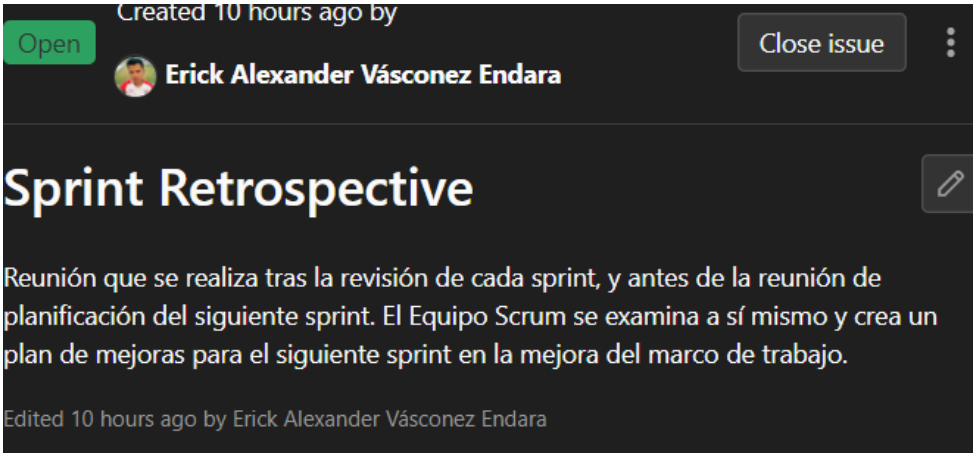


Figura 84 Sprint retrospective milestone 2

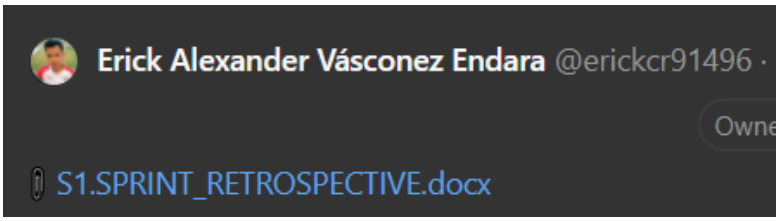


Figura 85 Recurso sprint retrospective milestone 2

Fuente: Propia

Iteración 3

Desarrollo del prototipado del Sprint

Sprint 3

Tabla 18 Prototipo Sprint 3

Sprint 3 **16/02/2022- 26/02/2022**		
Historia de Usuario	**Tarea**	**Horas**
Creación del módulo de características y	Diseño de la vista de formulario de checklist	5h

subcaracterísticas de la ISO 25012	Diseño y creación de la tabla evaluaciones relacionada con proyectos	15h
	CRUD de la tabla evaluaciones	10h
	Total	30h

Fuente: Propia

Gestión del proyecto con Boards

Sprint 3

- Issues del Sprint 3 (Milestone 3)

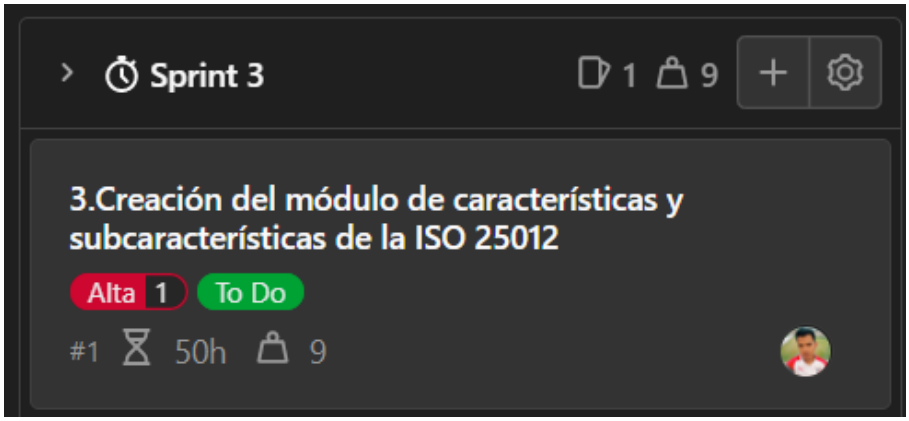


Figura 86 Board issue 3 milestone 3
Fuente: Propia

- Issues Por Hacer (To-Do)

Figura 87 Issue 3 to-do milestone 3

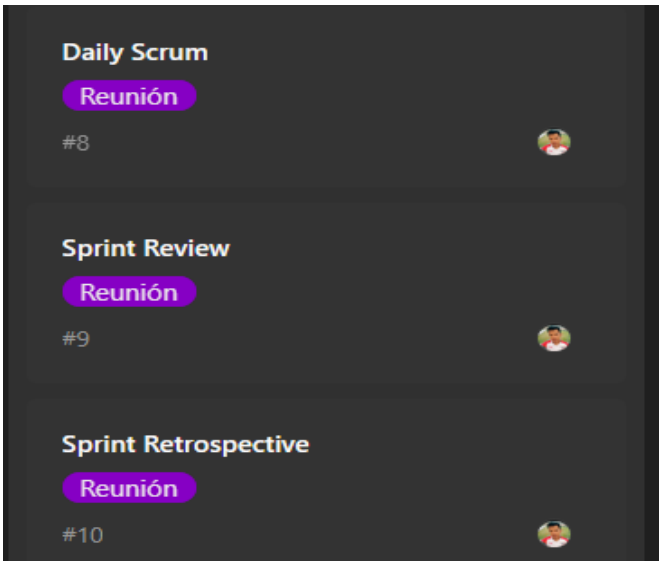


Figura 88 Reuiones milestone 3

Fuente: Propia

- Issues En Proceso (Doing)

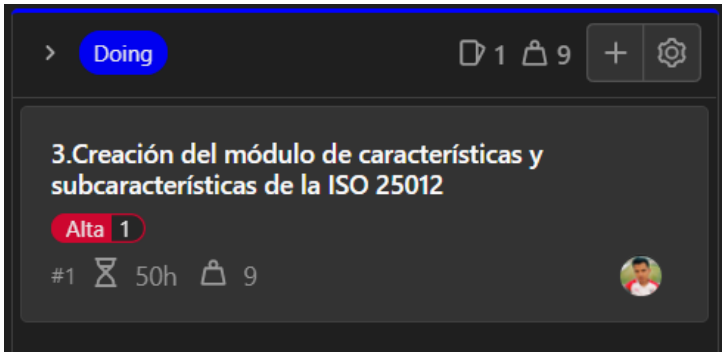


Figura 89 Issue 3 do-ing milestone 3
Fuente: Propia

Cierre del prototipo

- Issues Terminadas (Done)

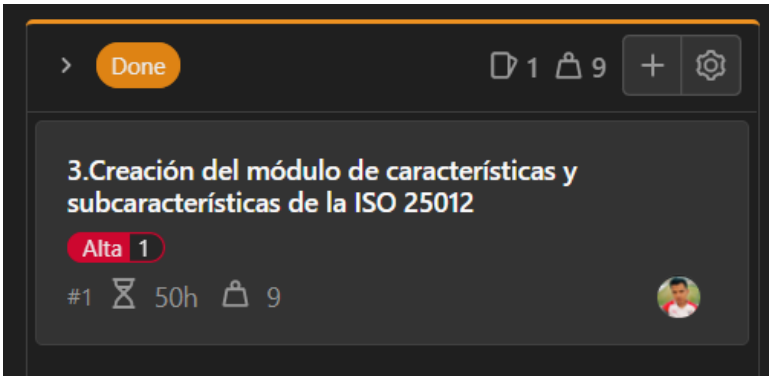


Figura 90 Issue 3 terminada done milestone 3
Fuente: Propia

- Issue cerrado en la columna Closed del Board

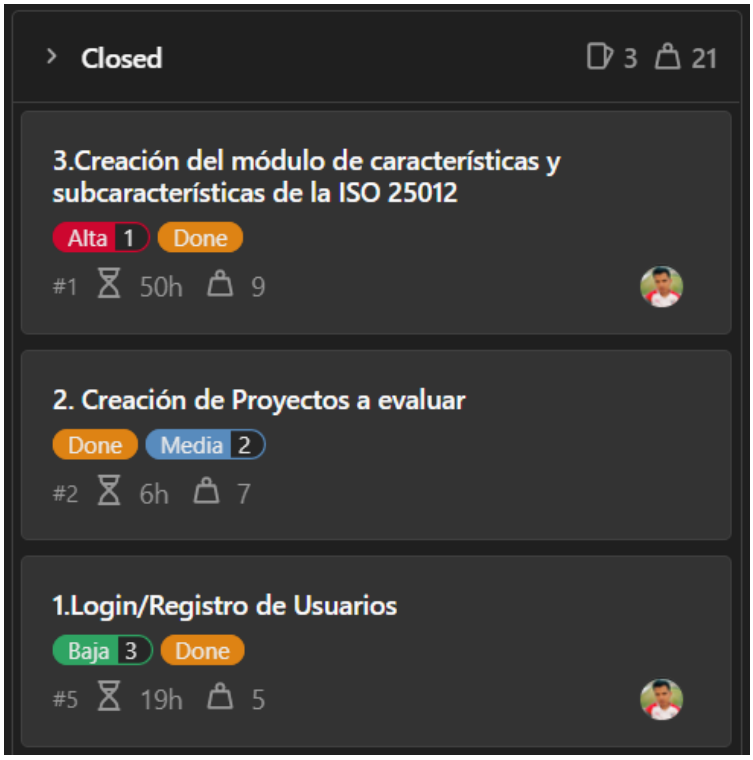


Figura 91 Issues board close milestone 3
Fuente: Propia

Daily Scrum

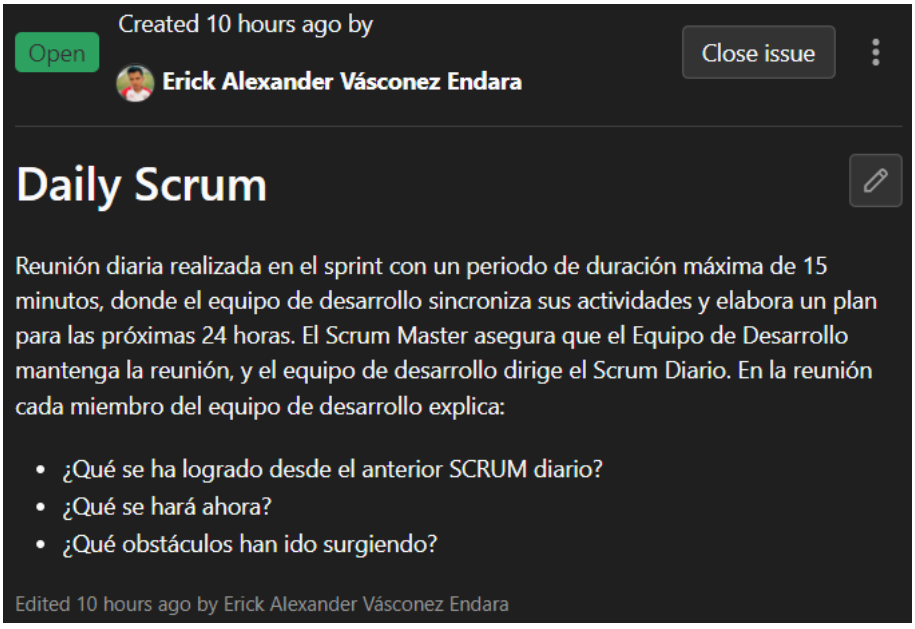


Figura 92 Daily Scrum milestone 3

Figura 93 Recurso Daily scrum milestone 3

Fuente: Propia

Sprint Review

Figura 94 Sprint review milestone 3

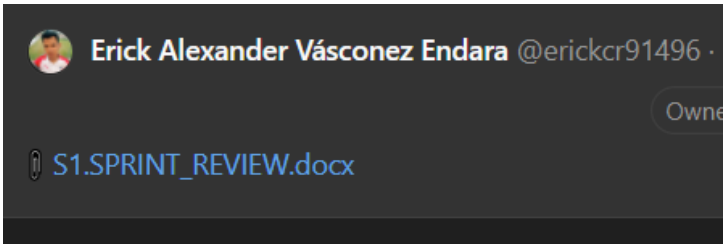


Figura 95 Recurso sprint review milestone 3

Fuente: Propia

Sprint Retrospective

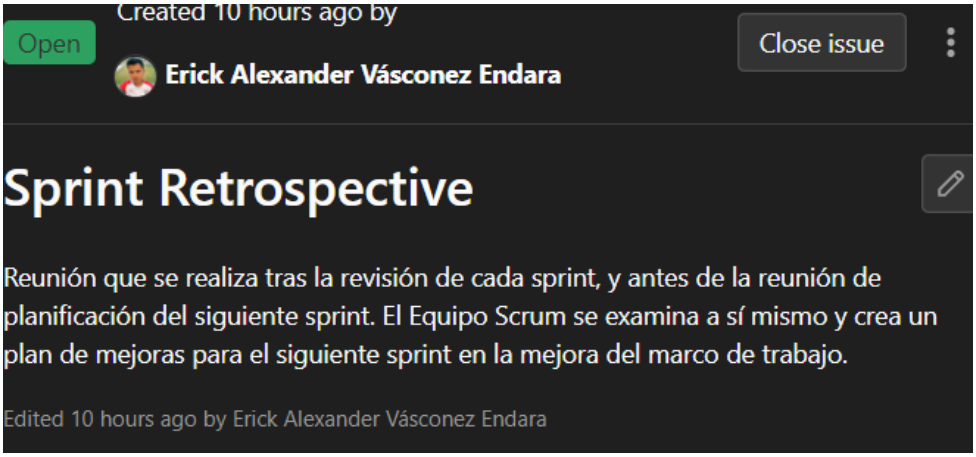


Figura 96 Sprint retrospective milestone 3

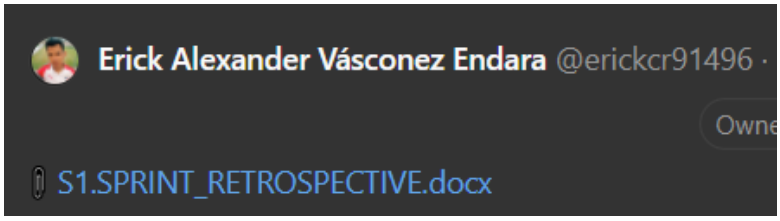


Figura 97 Recurso sprint retrospective milestone 3

Fuente: Propia

Iteración 4

Desarrollo del prototipado del Sprint

Sprint 4

Tabla 19 Prototipo Sprint 4

Sprint 4 **Fecha inicio- Fecha fin**		
Historia de Usuario	**Tarea**	**Horas**
Creación de reporte de evaluación	Diseño del reporte de evaluación detallado	5h
	Descargar reporte en pdf	5h

	Total	10h

Fuente: Propia

Gestión del proyecto con Boards

Sprint 4

- Issues del Sprint 4 (Milestone 4)

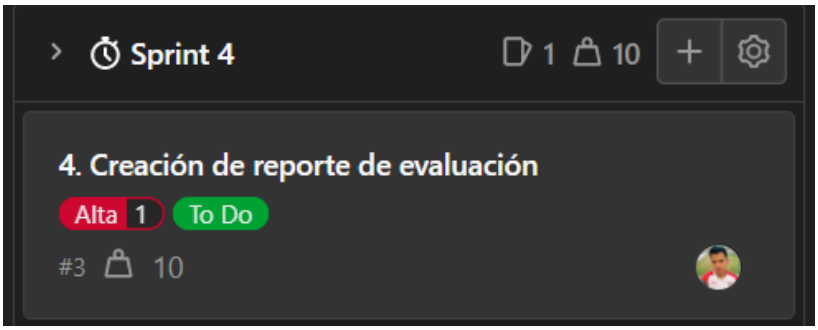


Figura 98 Board issue 4 milestone 4
Fuente: Propia

- Issues Por Hacer (To-Do)

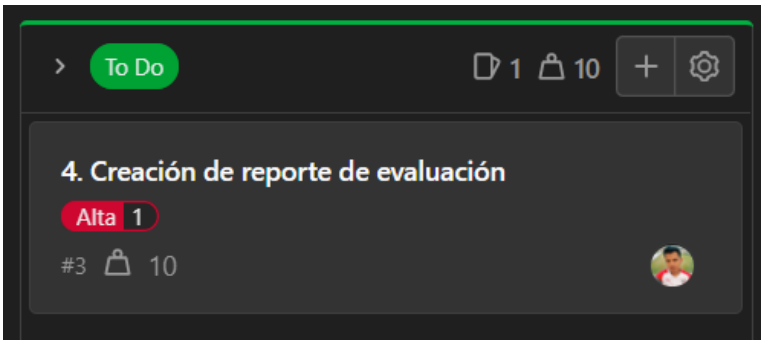


Figura 99 Issue 4 to-do milestone 4

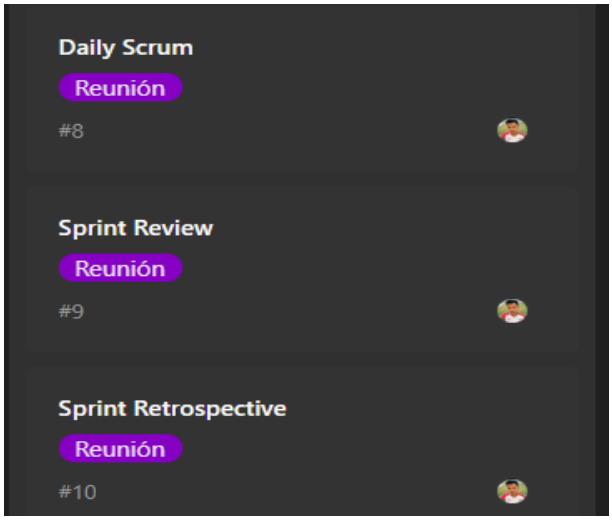


Figura 100 Reuniones milestone 4

Fuente: Propia

- Issues En Proceso (Doing)

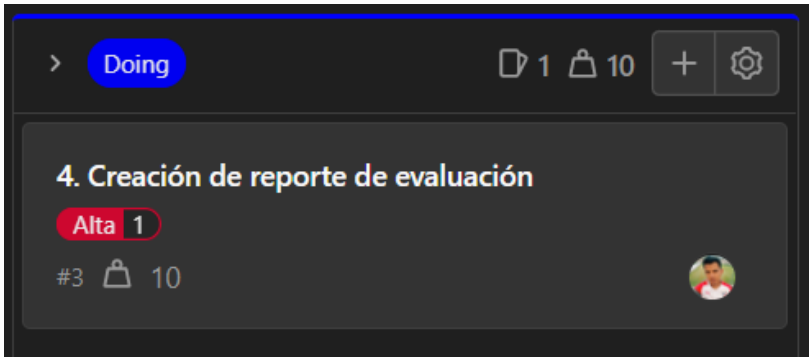


Figura 101 Issue 4 do-ing milestone 4

Cierre del prototipo

- Issues Terminadas (Done)

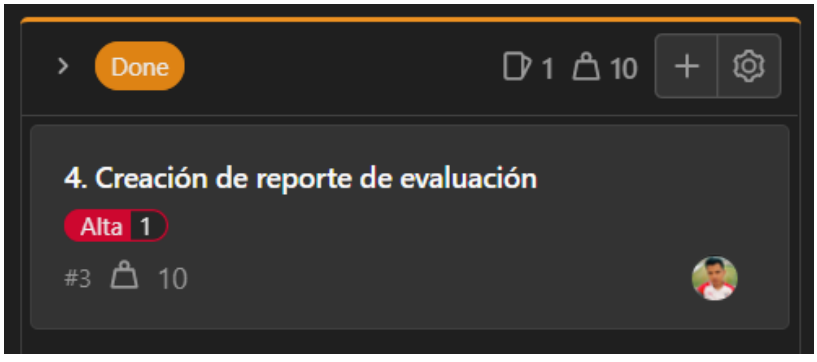


Figura 102 Issue 4 terminada done milestone 4
Fuente: Propia

- Issue cerrado en la columna Closed del Board

Figura 103 Board issue closed milestone 4
Fuente: Propia

Daily Scrum

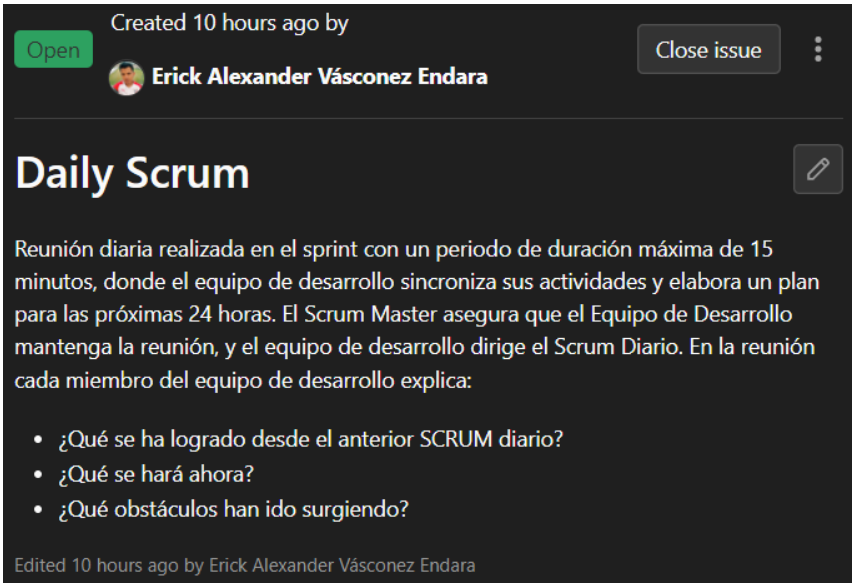


Figura 104 Daily scrum milestone 4

Figura 105 Recurso daily scrum milestone 4

Sprint Review

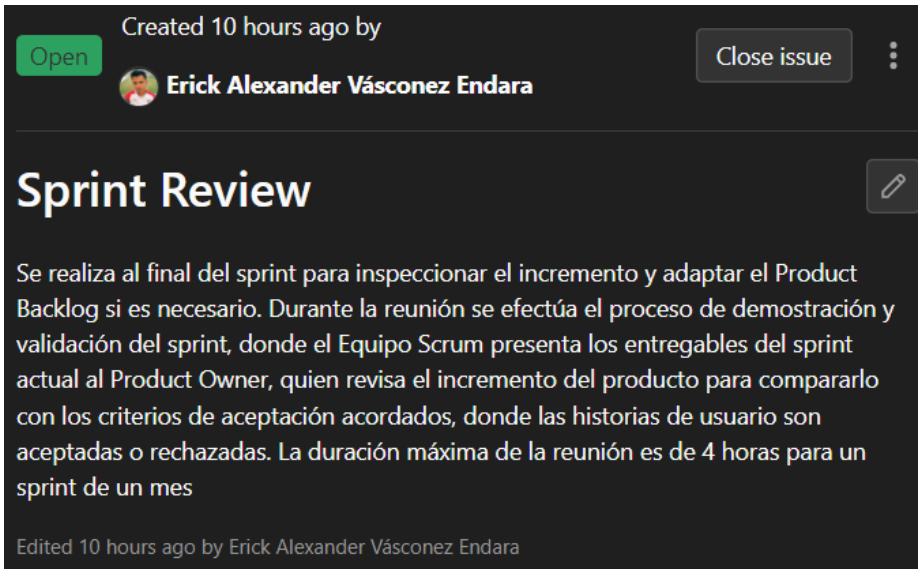


Figura 106 Sprint review milestone 4

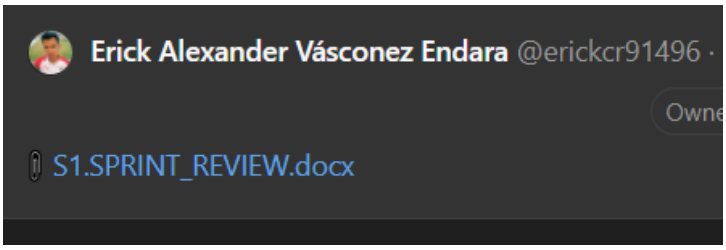


Figura 107 Recurso sprint review milestone 4

Fuente: Propia

Sprint Retrospective

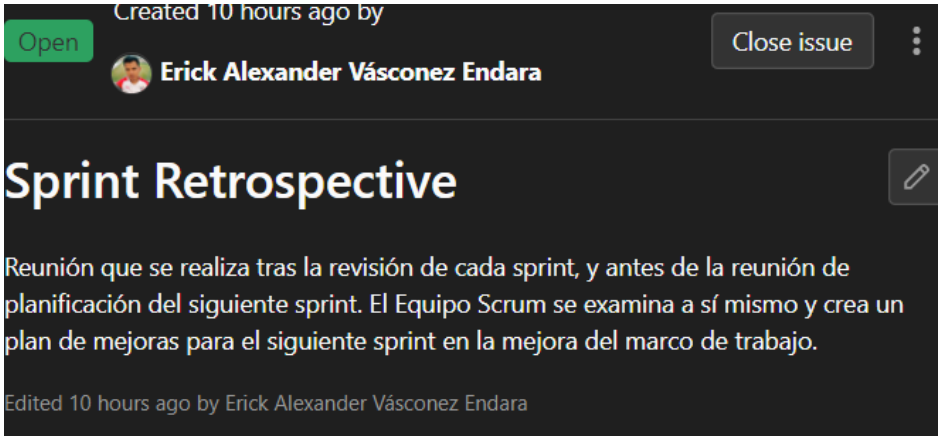


Figura 108 Sprint retrospective milestone 4

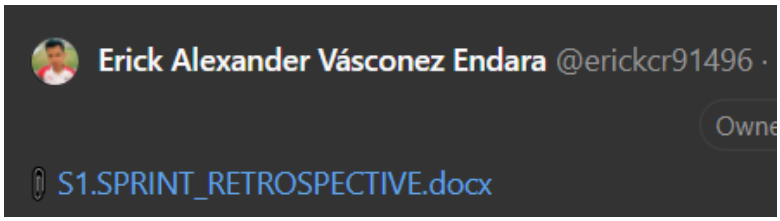


Figura 109 Recurso sprint retrospective milestone 4

2.4.3. Fase 3. Post-Juego

3.1 Incremento

Incremento Iteración 1

Etiquetar lanzamiento (Tag)

Se ha etiquetado como Release_v1.0.0 para nuestro incremento de sprint 1

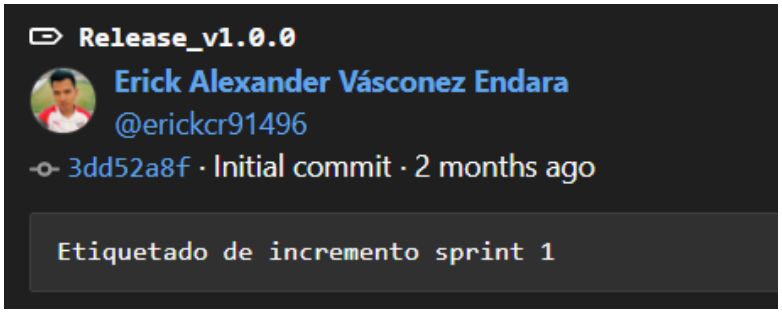


Figura 110 Tag incremento milestone 1
Fuente: Propia

Releases

Se definió el nombre para este reléase del incremento sprint 1 como **Incremento Sprint 1**, adjunto con las notas de la versión, metadatos, código fuente y el link de la aplicación puesta en producción.

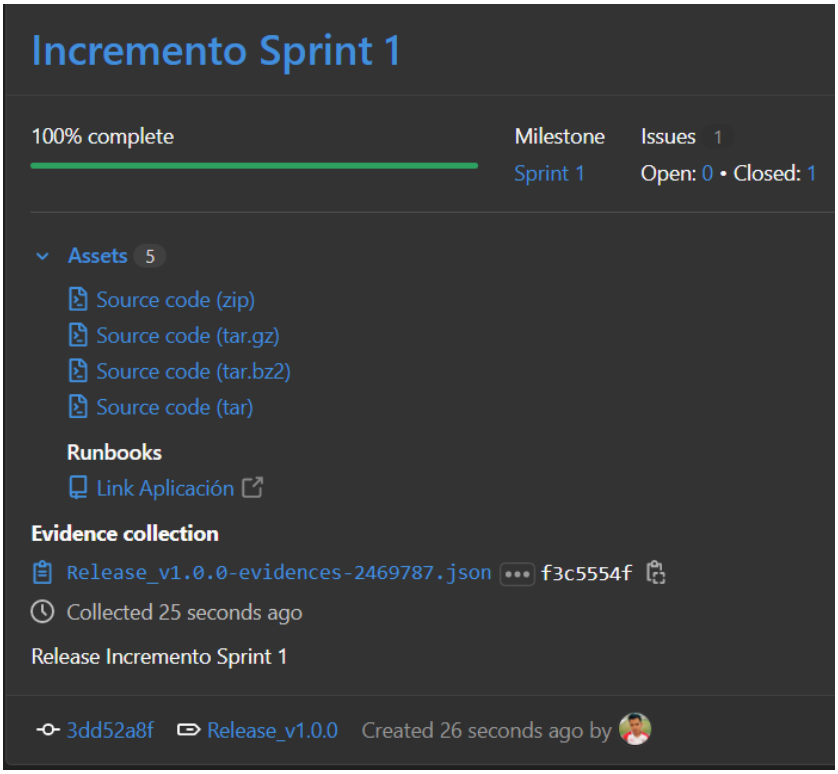


Figura 111 Release incremento milestone 1
Fuente: Propia

Incremento Iteración 2

Etiquetar lanzamiento (Tag)

Se ha etiquetado como Release_v2.0.0 para nuestro incremento de sprint 2

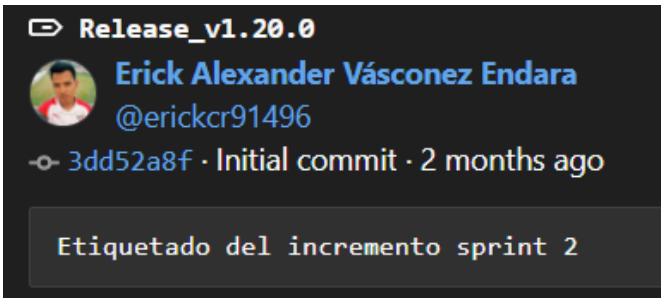


Figura 112 Tag incremento milestone 2
Fuente: Propia

Releases

Se definió el nombre para este reléase del incremento sprint 2 como **Incremento Sprint 2**, adjunto con las notas de la versión, metadatos, código fuente y el link de la aplicación puesta en producción.

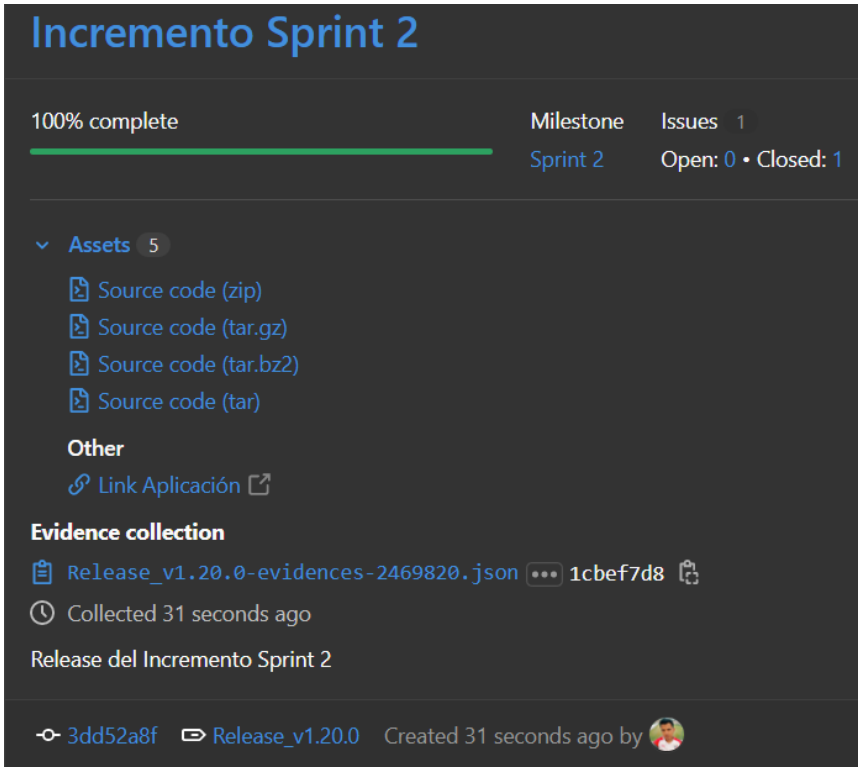


Figura 113 Release incremento milestone 2
Fuente: Propia

Incremento Iteración 3

Etiquetar lanzamiento (Tag)

Se ha etiquetado como Release_v3.0.0 para nuestro incremento de sprint 3

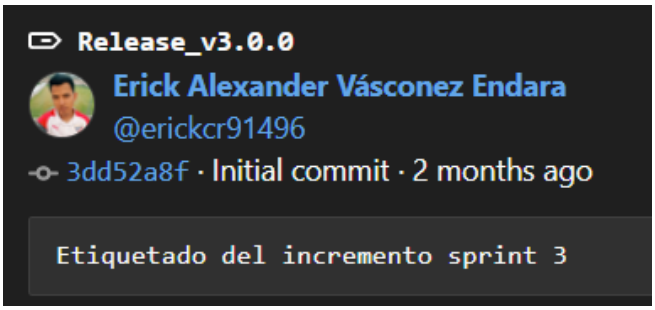


Figura 114 Tag incremento milestone 3
Fuente: Propia

Releases

Se definió el nombre para este reléase del incremento sprint 3 como **Incremento Sprint 3**, adjunto con las notas de la versión, metadatos, código fuente y el link de la aplicación puesta en producción.

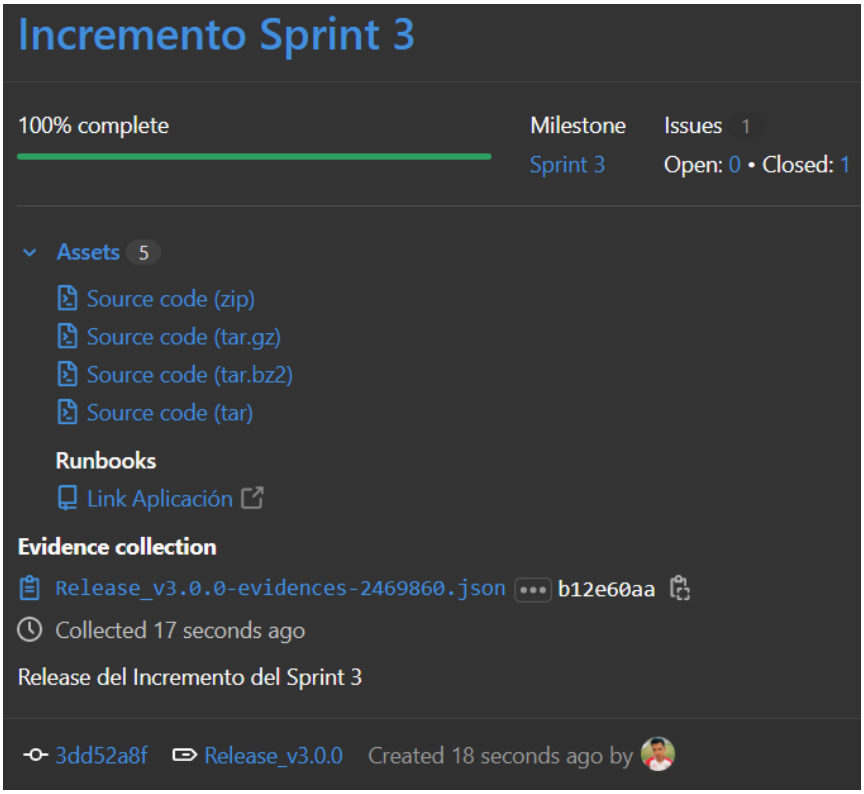


Figura 115 Release incremento 3 milestone 3
Fuente: Propia

Incremento Iteración 4

Etiquetar lanzamiento (Tag)

Se ha etiquetado como Release_v4.0.0 para nuestro incremento de sprint 4

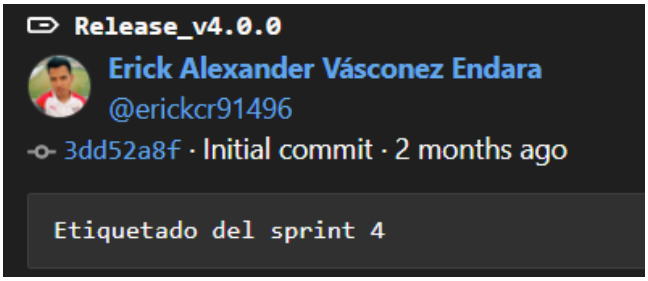


Figura 116 Tag incremento 4 milestone 4
Fuente: Propia

Releases

Se definió el nombre para este reléase del incremento sprint 4 como **Incremento Sprint 4**, adjunto con las notas de la versión, metadatos, código fuente y el link de la aplicación puesta en producción.

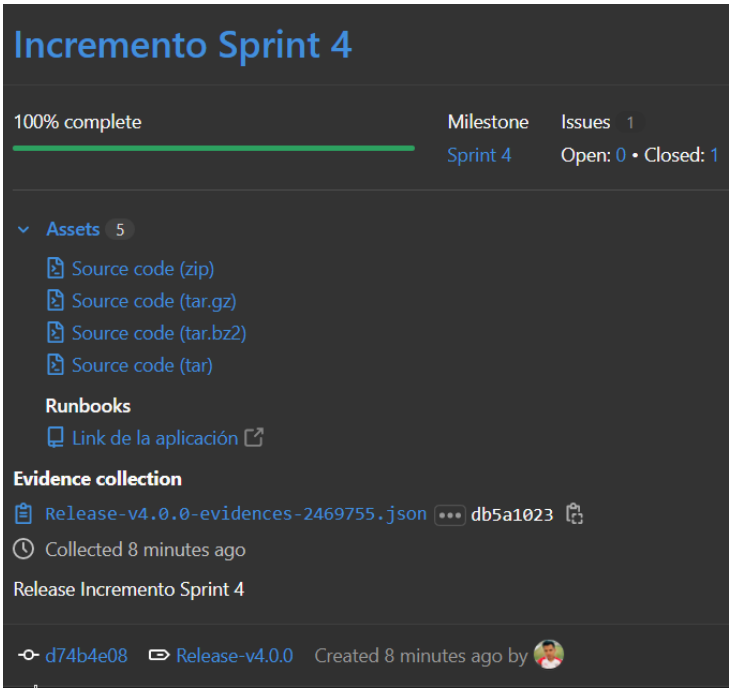


Figura 117 Release incremento 4 milestone 4
Fuente: Propia

CAPÍTULO III

Resultados

3. Validación de la guía metodológica

3.1. Metodología

3.1.1. Escala de Likert

Tabla 20 Escala de likert

Totalmente en desacuerdo	En desacuerdo	Ni acuerdo ni en desacuerdo	De acuerdo	Totalmente de acuerdo
1	2	3	4	5

Fuente: Propia

3.2. Éxito de cumplimiento de la guía

Para la validación de la guía metodológica se desarrolló la prueba de concepto aplicando las actividades propuestas por la guía metodológica, en las cuales se obtuvo los siguientes resultados.

Tabla 21 Nivel de cumplimiento guía metodológica

Fase	Actividades	Éxito de cumplimiento (1-5)	Observaciones
1. PRE - JUEGO	**1.1** Inicio del proyecto	5	
	1.2 Definición de roles de scrum	5	
	1.3 Definición del Product Backlog	5	
	1.4 Creación de milestones	3	Este paso se lo puede haber creado también en la fase de Juego.
	1.5 Definición de tareas de Historia de Usuario	3	El Time-tracking no afecta en el desarrollo del issue, solo es un indicador de avance del tiempo estimado

			con el tiempo empleado	
2. JUEGO O DESARROLLO	**2.1** Sprint Planning	4	Puede ser considerado también dentro del paso 2.3	
	2.2 Sprint Backlog	5		
	2.3 Iteraciones del prototipado	5		
3. POST- JUEGO	**3.1** Incremento	5		
	Total	**40**	Porcentaje de éxito	**%**

Fuente: Propia

El total de puntos de la guía metodológica en el mejor de los casos es de **45** para un porcentaje de éxito del 100%, al seguir los pasos con la prueba de concepto puntuamos cada una de las actividades y al sumar los todos los puntos se obtuvo un total de 40 puntos con un porcentaje de éxito del **88.8 %.**

3.1 Encuesta de usabilidad

El cuestionario de Escala de Usabilidad del Sistema, o System Usability Scale (SUS) original consta de cinco preguntas positivas y cinco negativas, mediante una escala de 5 niveles que van de “Totalmente en desacuerdo” a “Totalmente de acuerdo” (Hedlefs Aguila & Garza Villegas, 2016).

En base a esto, se realizará una encuesta mediante Microsoft Forms que consta con un cuestionario de diez preguntas en una adaptación de este cuestionario al español de (Hedlefs Aguila & Garza Villegas, 2016) en su versión positiva, tal como recomiendan los autores de esta adaptación.

El cuestionario consta de las siguientes diez preguntas:

1. ¿Cree que le gustaría utilizar este sistema con frecuencia?
2. ¿Encontró el sitio web sencillo?
3. ¿Piensa que el sitio web es fácil de usar?

4. ¿Piensa que podría utilizar este sitio web sin el apoyo de un técnico?
5. ¿Encontró que varias de las funciones en el sitio web estaba bien integradas?
6. ¿Piensa que había demasiada inconsistencia en el sistema?
7. ¿Imagina que la mayoría de la gente aprendería a usar este sistema de forma muy rápida?
8. ¿Encontró el sitio web difícil de usar?
9. ¿Se sintió confiado (seguro) al utilizar el sitio web?
10. ¿Puede utilizar el sitio web sin tener que aprender algo nuevo?

Para evaluar la encuesta se ha tomado en cuenta la escala de Likert de 5 puntos siendo 5 totalmente de acuerdo y 1 totalmente en desacuerdo (Sauro & Lewis, 2016).

1. Totalmente en desacuerdo
2. En desacuerdo
3. Ni de acuerdo ni en desacuerdo
4. De acuerdo
5. Totalmente de acuerdo

Los resultados del cuestionario se los representa en la tabla de la siguiente manera, en la primera columna las opciones o posibles respuestas y de ahí hacia la derecha cada columna enumerada del 1 al 10 haciendo referencia a cada una de las preguntas descritas elaboradas en el cuestionario.

Tabla 22 Total de votos por preguntas

Respuesta (opciones)	1	2	3	4	5	6	7	8	9	10
Totalmente en desacuerdo	0	0	0	0	0	7	0	9	1	0
En desacuerdo	0	1	0	2	0	14	1	8	2	3
Ni de acuerdo ni en desacuerdo	7	3	4	4	4	3	4	4	4	2
De acuerdo	16	9	12	10	18	2	17	5	11	12
Totalmente de acuerdo	7	17	14	14	8	4	8	4	12	13
Total	30	30	30	30	30	30	30	30	30	30

Fuente: Propia

3.2 Análisis de resultados

A continuación, se detallan los resultados obtenidos de cada pregunta de la encuesta SUS aplicada a 30 estudiantes de la carrera de Software.

Pregunta 1 ¿Cree que le gustaría utilizar este sistema con frecuencia?

La opción más votada fue “De acuerdo” con el 54% de los votos, seguido de las opciones “Totalmente de acuerdo” y “Ni acuerdo ni en desacuerdo” con un 23% cada una. Eso indica que más de la mitad volverían a utilizar este sistema.

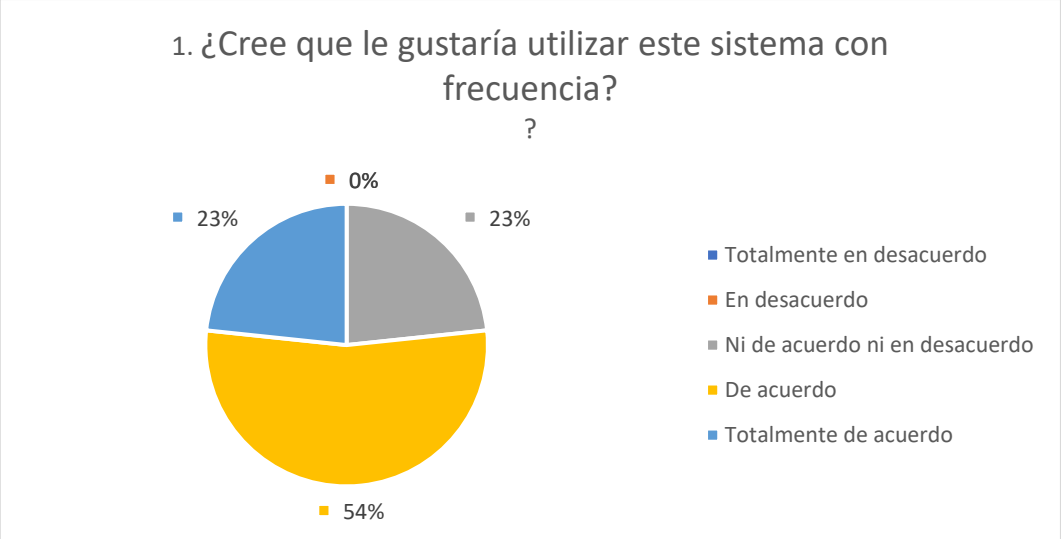


Figura 118 Respuestas pregunta 1
Fuente: Propia

Pregunta 2: ¿Encontró el sitio web sencillo?

La opción más votada fue “Totalmente de acuerdo” con el 57% de los votos, seguido de la opción “De acuerdo” con el 30%. Esto indica que el usuario no encuentra complejidad al navegar por el sitio web.

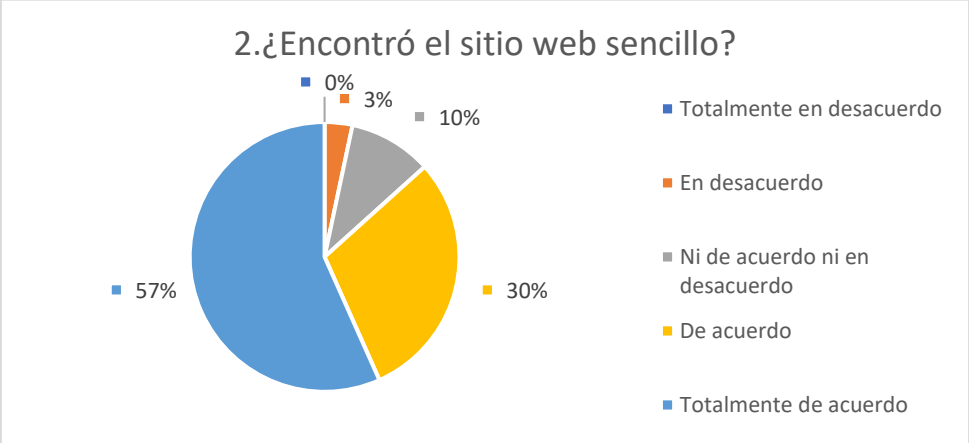


Figura 119 Resputas pregunta 2
Fuente: Propia

Pregunta 3: ¿Piensa que el sitio web es fácil de usar?

La opción más votada fue “Totalmente de acuerdo” con el 47% de los votos, seguido de la opción “De acuerdo” con el 40%. Esto indica que el usuario afirma que el sitio web no tiene dificultad de uso.

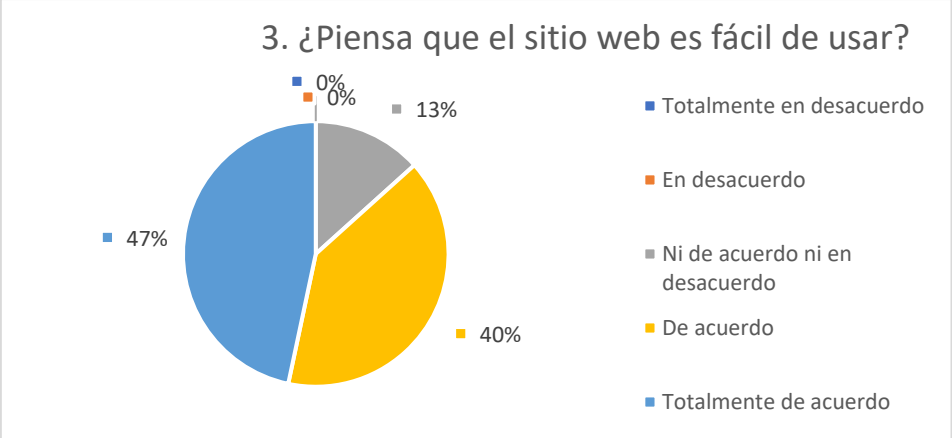


Figura 120 Respuestas pregunta 3
Fuente: Propia

Pregunta 4: ¿Piensa que podría utilizar este sitio web sin el apoyo de un técnico?

La opción más votada fue “Totalmente de acuerdo” con el 47% de los votos, seguido de la opción “De acuerdo” con el 33%. Los resultados indican que el usuario no necesita de poca o ninguna ayuda para usar el sitio web.

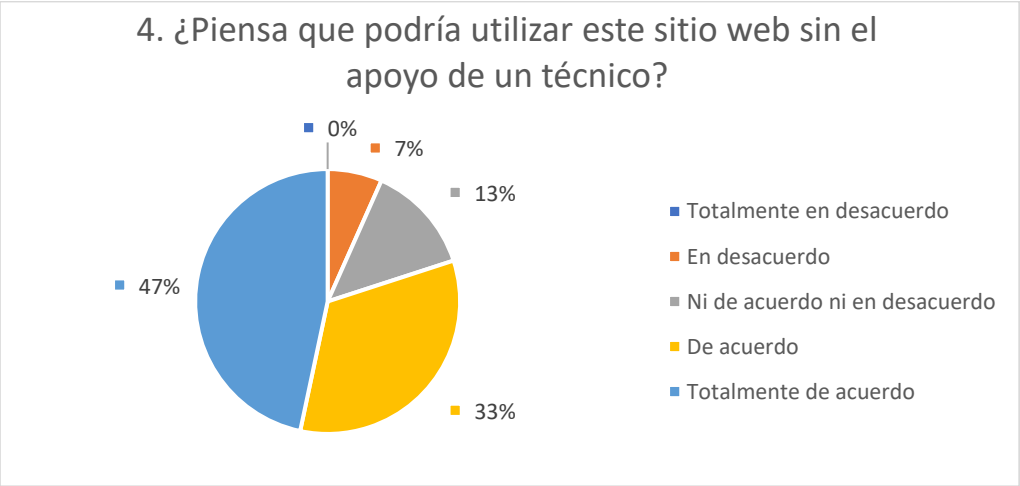


Figura 121 Respuestas pregunta 4
Fuente: Propia

Pregunta 5: ¿Encontró que varias de las funciones en el sitio web estaba bien integradas?

La opción más votada fue “De acuerdo” con el 60% de las respuestas, seguido de la opción “Totalmente de acuerdo” con el 27%. Eso indica que el usuario encuentra la estructura del funcionamiento en orden y sin problemas.

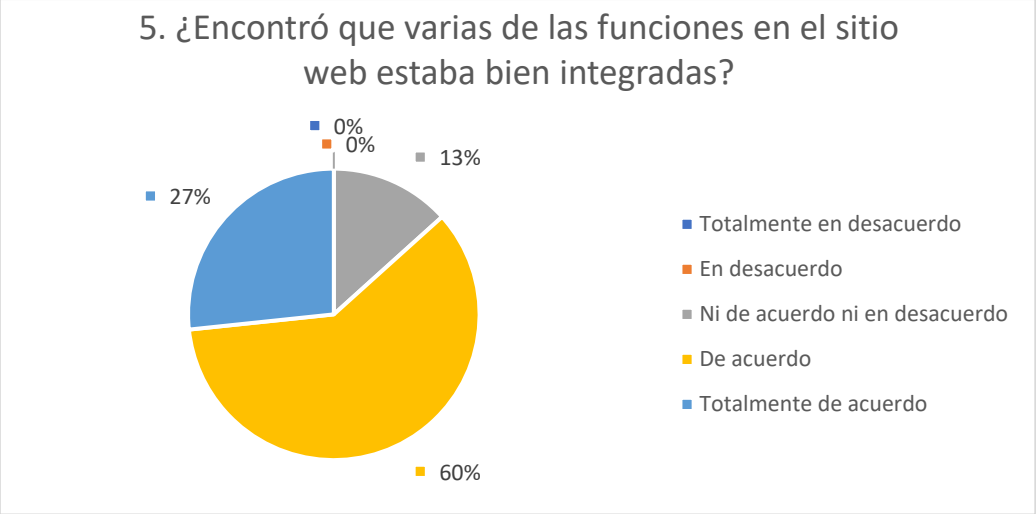


Figura 122 Respuestas pregunta 5
Fuente: Propia

Pregunta 6: ¿Piensa que había demasiada inconsistencia en el sistema?

La opción más votada fue “En desacuerdo” con el 47% de los votos seguido de la opción “Totalmente en desacuerdo” con el 23%. Los resultados indican que la consistencia entre las funcionalidades del sistema tiene congruencia.

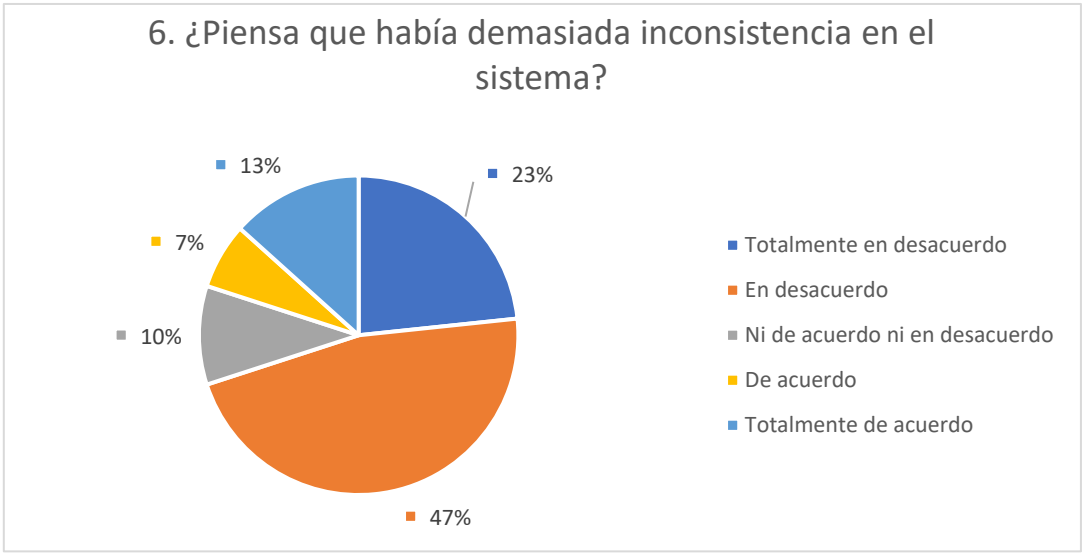


Figura 123 Respuestas pregunta 6
Fuente: Propia

Pregunta 7: ¿Imagina que la mayoría de la gente aprendería a usar este sistema de forma muy rápida?

La opción más votada fue “De acuerdo” con el 57% de los votos seguido de la opción “Totalmente de acuerdo” con el 27%. Los resultados indican que el usuario no encuentra dificultad ni requiere de mucho aprendizaje para utilizar el sistema.

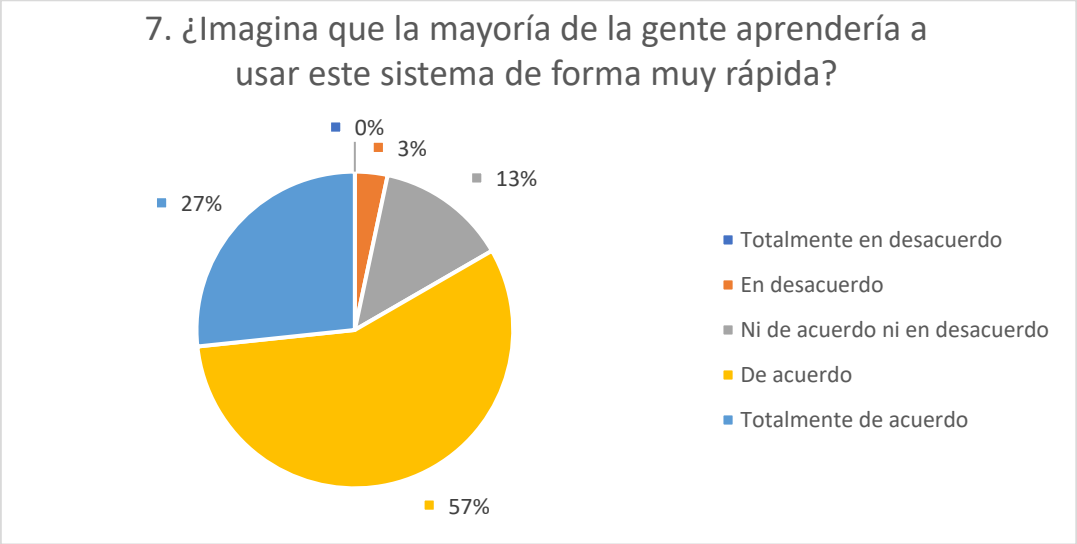


Figura 124 Respuestas pregunta 7
Fuente: Propia

Pregunta 8: ¿Encontró el sitio web difícil de usar?

La opción más votada fue “Totalmente en desacuerdo” con el 30% de los votos seguido de la opción “En desacuerdo” con el 27%. Los resultados afirman que el usuario encuentra el sistema muy intuitivo y tiene claro las actividades que quiere realizar dentro del mismo.

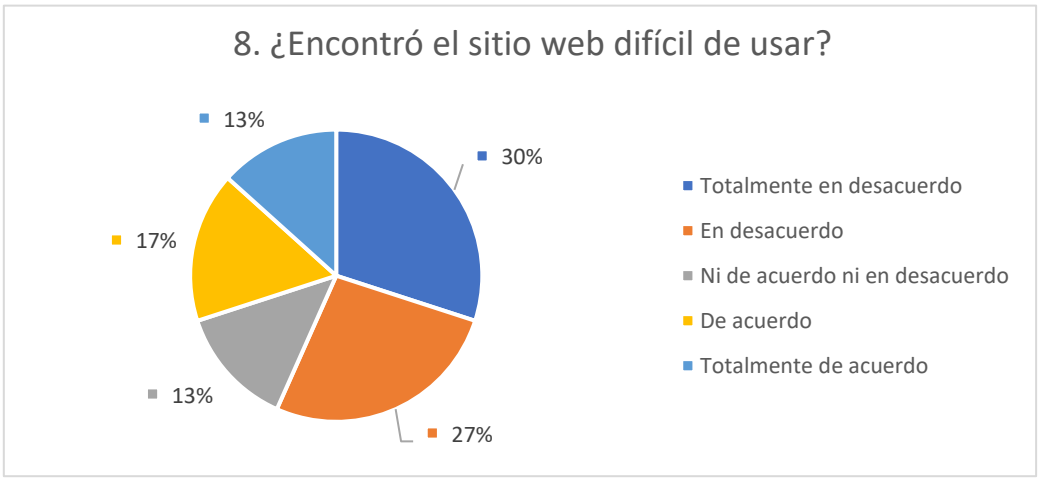


Figura 125 Respuestas pregunta 8
Fuente: Propia

Pregunta 9: ¿Se sintió confiado (seguro) al utilizar el sitio web?

La opción más votada fue “Totalmente de acuerdo” con el 40% de los votos seguido de la opción “De acuerdo” con el 37%. Los resultados afirman que el usuario se siente confiado y seguro mientras usan el sistema.

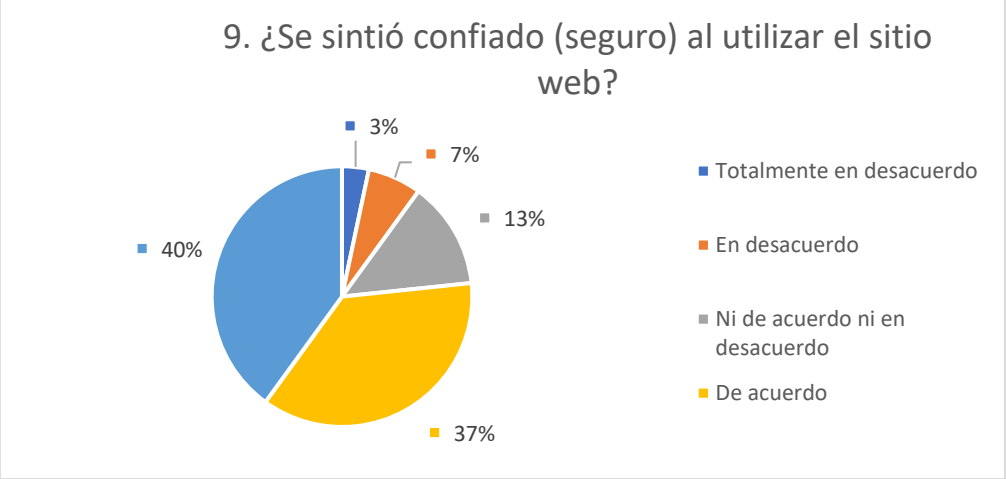


Figura 126 Respuestas pregunta 9
Fuente: Propia

Pregunta 10: ¿Puede utilizar el sitio web sin tener que aprender algo nuevo?

La opción más votada fue “Totalmente de acuerdo” con el 43% de los votos seguido de la opción “De acuerdo” con el 40%. Esto indica que el uso del sistema es fácil y no se requiere mucho conocimiento.

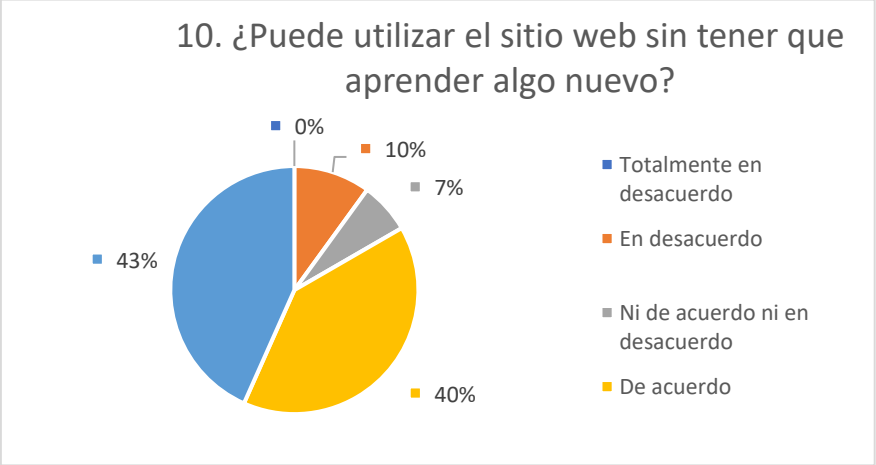


Figura 127 Respuestas pregunta 10
Fuente: Propia

CONCLUSIONES

La guía elaborada se la realizo con el fin de dejar una pauta para futuras investigaciones que permitan mejorar la gestión de proyectos en el desarrollo de software por medio de distintas herramientas y/o plataformas de gestión, es decir que esta guía será un material de apoyo para próximos proyectos que apliquen metodología Scrum.

Se establecieron diferentes modelos de guías y manuales para la selección del esquema central de la guía, la estructura de guía presentada por Guzmán fue la seleccionada para este trabajo debido a que contiene los formatos y diagramas de flujo que son necesarios para desarrollar la guía.

La guía metodológica se constituye como una herramienta integral para la gestión de proyectos de software, combinando las mejores prácticas de Scrum administradas por GitLab estableciendo prácticas ágiles en tres fases principales que contiene la guía que son: Pre-juego, juego y post-juego.

La guía metodológica tiene un total de 9 actividades divididas en 3 fases con un total de 45 puntos, y al aplicar la guía metodológica con la prueba de concepto se obtuvo un porcentaje de éxito del 90%, el cual ratifica la aceptación de la guía metodológica.

RECOMENDACIONES

Para los futuros trabajos de investigación encaminados a realizar técnicas de elaboración de una guía tomar en cuenta distintos modelos o estructuras existentes y seleccionar el más conveniente de acuerdo con las necesidades.

Se recomienda continuar con nuevas versiones a la guía propuesta con el fin de mejorar el proceso de la gestión de proyectos con Scrum y cualquier otra herramienta que permita realizar gestión de proyectos de software.

Se recomienda al equipo de trabajo e interesados mantener una comunicación efectiva que permita identificar nuevos puntos de mejora de la guía para complementarla y hacerla aún más robusta.

Se debe aclarar que la metodología Scrum se aplica para proyectos ágiles y no al tradicional modelo en cascada (waterfall), para no generar malentendidos durante la entrega o incremento del producto como lo establece Scrum.

REFERENCIAS

Almeida, H., & Correia, W. (2019). Agile Project Management: Better Deliveries to the End User in Software Projects with a Management Model by Scrum. In *Advances in Intelligent Systems and Computing* (Vol. 794, pp. 293–305). Springer Verlag. https://doi.org/10.1007/978-3-319-94947-5_29

Cabral, A. R., Ribeiro, M. B., & Noll, R. P. (2014). Knowledge Management in Agile Software Projects: A Systematic Review. *Journal of Information & Knowledge Management*, *13*(01), 1450010. https://doi.org/10.1142/S0219649214500105

Carrizo, D., Moller, C., Carrizo, D., & Moller, C. (2018). Estructuras metodológicas de revisiones sistemáticas de literatura en Ingeniería de Software: un estudio de mapeo sistemático. *Ingeniare. Revista Chilena de Ingeniería*, *26*, 45–54. https://doi.org/10.4067/S0718-33052018000500045

Chamorro, L. (2018). *IMPLEMENTACIÓN DE UNA METODOLOGÍA DE DESARROLLO ÁGIL PARA EL MEJORAMIENTO DE LA GESTIÓN DE LOS PROYECTOS DE SOFTWARE EN LA UNIVERSIDAD DE INVESTIGACIÓN DE TECNOLOGÍA EXPERIMENTAL YACHAY*. http://repositorio.utn.edu.ec/bitstream/123456789/8973/2/PG%20715%20TRABAJO%20DE%20GRADO.pdf

Diaz, A. M., Casañola, Y. T., & Hidalgo, D. B. (2019). Notes to manage quality activities in software development projects to reduce the costs of correcting defects. *Ingeniare*, *27*(2), 319–327. https://doi.org/10.4067/S0718-33052019000200319

Diéguez, M., & Cares, C. (2011). *De la Gestión de Seguridad en el Ciclo de Vida del Software*.

Guerrero, G., & Guevara, A. (2019). *INTEGRACIÓN DEL MODELO CMMI-DEV Y EL MARCO DE TRABAJO SCRUM, EN EL PROCESO DE DESARROLLO DE SOFTWARE*. http://repositorio.utn.edu.ec/bitstream/123456789/9972/2/PG 776 TRABAJO GRADO.pdf

Guzman, E. (2016). *PROPUESTA METODOLÓGICA USANDO SCRUM Y PMBOK, PARA LA GESTIÓN DE PROYECTOS DE TI DE LA JEFATURA DE*

INFORMÁTICA DE UNA UNIDAD EJECUTORA DEL SECTOR TRANSPORTES. https://doi.org/10.13140/RG.2.2.20323.96803

Hayat, F., Rehman, A. U., Arif, K. S., Wahab, K., & Abbas, M. (2019). The Influence of Agile Methodology (Scrum) on Software Project Management. *2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 145–149. https://doi.org/10.1109/SNPD.2019.8935813

Hedlefs Aguila, & Garza Villegas. (2016). *Adaptación al español del Cuestionario de Usabilidad de Sistemas Informáticos CSUQ*. https://www.reci.org.mx/index.php/reci/article/view/35/116

Iriz Ricote, S. (2019). *Seguridad en el ciclo de vida del desarrollo del software: DevSecOps*. http://hdl.handle.net/10609/96129

Lehtovirta, N. M. (2017). *Managing software project with GitLab How it is done at Sparta Consulting Oy*. https://www.theseus.fi/bitstream/handle/10024/138975/opinnaytetyo_niko_lehtovirta.pdf?sequence=1

Ludeña, H. (2018). *Diseño de una guía para el desarrollo de aplicaciones móviles* [Universidad de las Fuerzas Armadas ESPE. Maestría en Gerencia de Sistemas.]. http://repositorio.espe.edu.ec/jspui/handle/21000/15252

Mikko Dittmer. (2019). *GitLab Project management* . https://www.theseus.fi/bitstream/handle/10024/172491/Dittmer_Mikko.pdf?sequence=2

Ordoñez Ortiz, J. F., & Valle Caicedo, L. E. (2014). *Diseño de un marco de referencia para el emprendimiento de negocios de TI basado en la filosofía "Lean Startup"* [Quito: Universidad de las Américas, 2014]. http://dspace.udla.edu.ec/handle/33000/3059

Patricia, L., & Guerrero, C. (2016). Gestión en Proyectos de Software. *Tecnología Investigación y Academia*, *4*(2), 12–19.

Rea, R. (2017). *Guía metodológica para la gestión de requerimientos de desarrollo de software del Gobierno Autónomo Descentralizado Municipal de San Miguel de Ibarra*. http://repositorio.utn.edu.ec/handle/123456789/7688

Renato, A. (2016). *DESARROLLO DE UNA GUIA DE UN MARCO DE REFERENCIA DE CALIDAD PARA LA METODOLOGÍA DE DESARROLLO DE SOFTWARE ÁGIL SCRUM*.

Sauro, J., & Lewis, J. R. (2016). Standardized usability questionnaires. In *Quantifying the User Experience* (pp. 185–248). Elsevier. https://doi.org/10.1016/b978-0-12-802308-2.00008-4

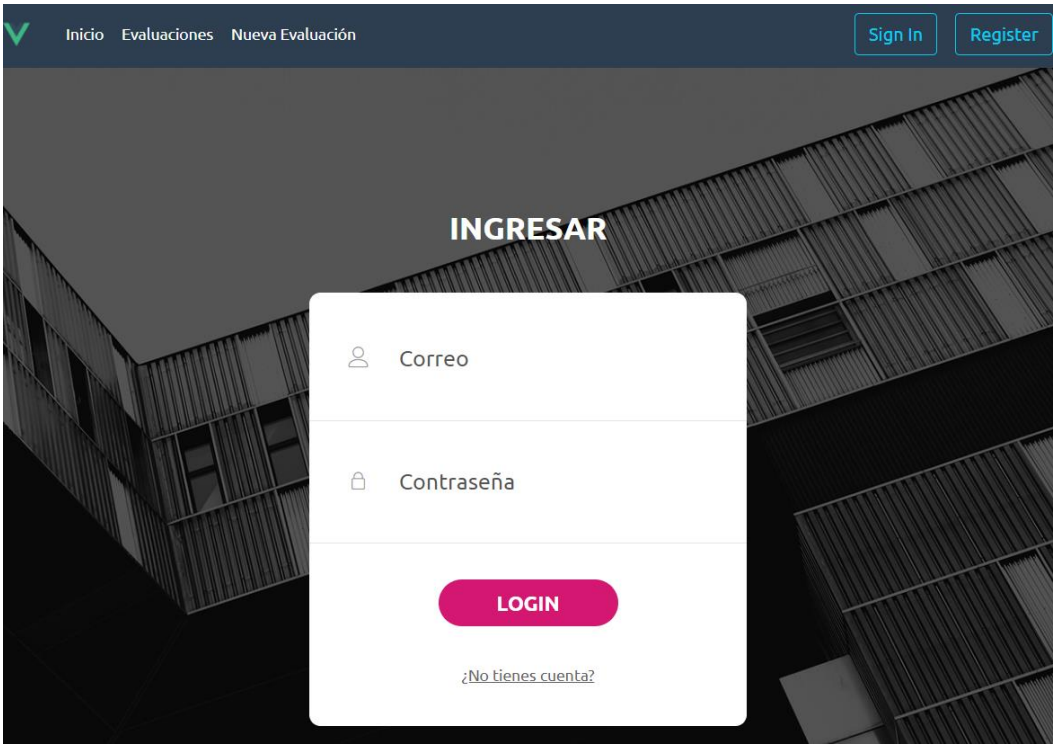
Schwaber, K., & Sutherland, J. (2017). *La Guía de Scrum*. https://scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Spanish-SouthAmerican.pdf

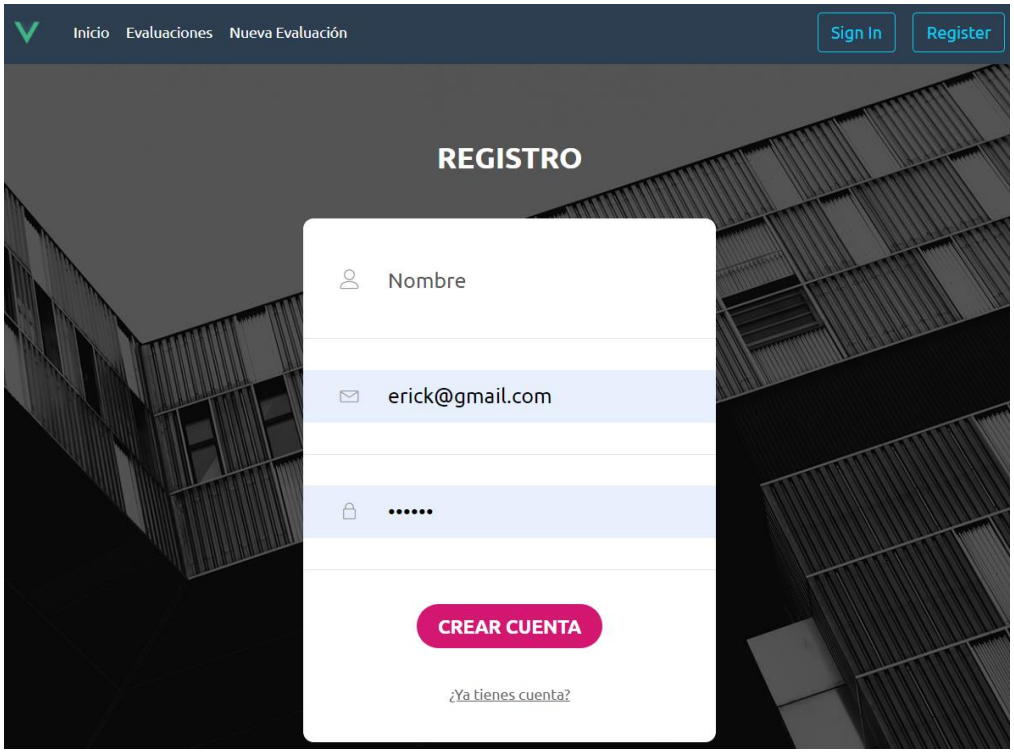
Soria José. (2021). *Desarrollo de un repositorio de versionamiento mediante la herramienta gitlab para almacenar de código fuente de los sistemas de software en la dirección de tecnologías de la información y comunicación del ejército ecuatoriano* [Latacunga: Universidad de la Fuerzas Armadas ESPE, 2021]. http://repositorio.espe.edu.ec/jspui/handle/21000/25951

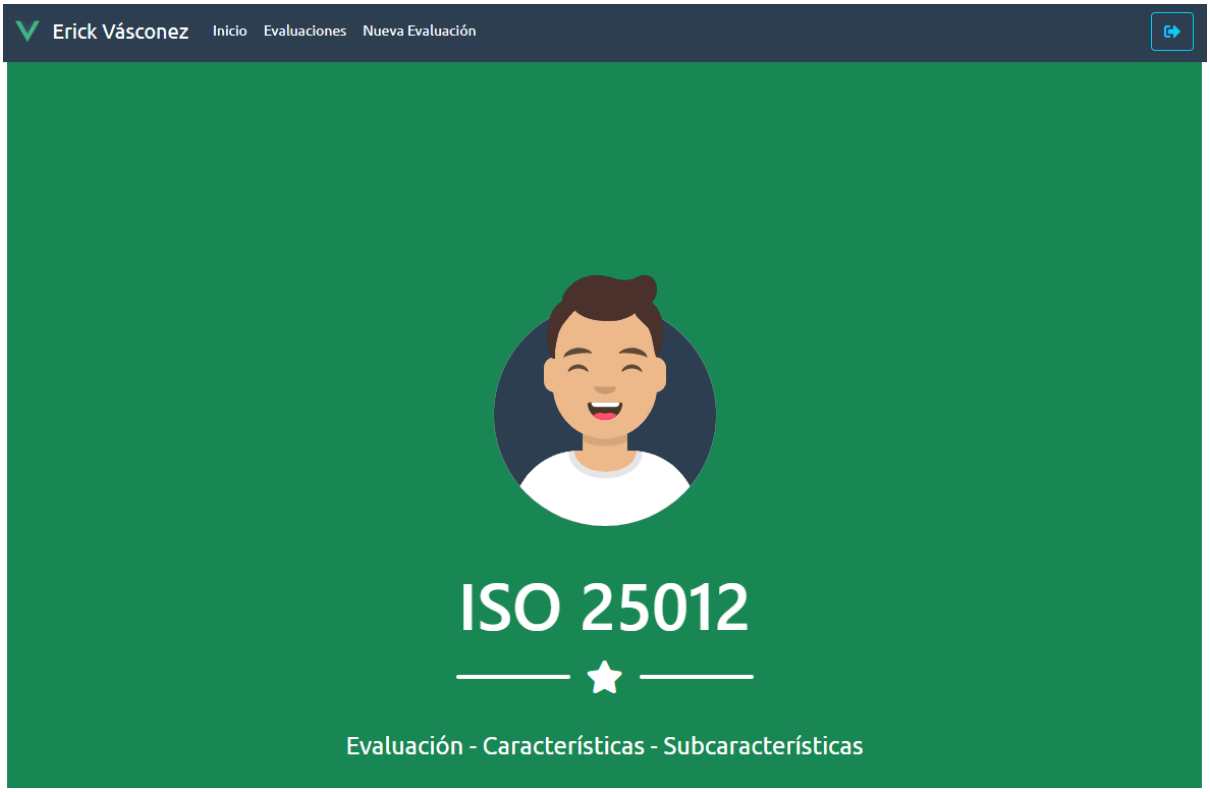
Usability.gov. (2013). *System Usability Scale (SUS)*. https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

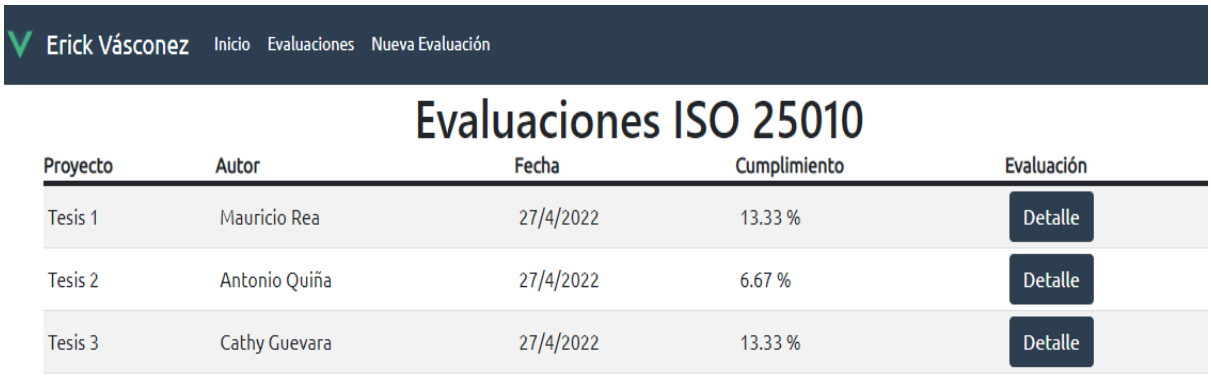
ANEXOS

Prueba de concepto






Erick Vásconez
Inicio
Evaluaciones
Nueva Evaluación
ISO 25012
Evaluación - Características - Subcaracterísticas


Erick Vásconez
Inicio
Evaluaciones
Nueva Evaluación
Evaluaciones ISO 25010
Proyecto
Autor
Fecha
Cumplimiento
Evaluación
Tesis 1
Mauricio Rea
27/4/2022
13.33 %
Detalle
Tesis 2
Antonio Quiña
27/4/2022
6.67 %
Detalle
Tesis 3
Cathy Guevara
27/4/2022
13.33 %
Detalle


Erick Vásconez
Inicio
Evaluaciones
Nueva Evaluación
VALIDACIÓN ISO 25012
DATOS DEL PROYECTO A EVALUAR
Nombre del proyecto
Nombre proyecto
Autor del proyecto
Autor
Características y Subcaracterísticas
Precisión de la información
1. Credibilidad :
Grado en el que los datos tienen atributos que permiten mantener y preservar un nivel específico de operaciones y calidad, incluso en caso de fallos, en un contexto de uso específico.
2. Actualidad:
Grado en el que los datos tienen atributos que tienen la edad correcta en un contexto de uso específico.
3. Precisión:
Grado en el que los datos tienen atributos que son exactos o proporcionan discernimiento en un contexto de uso específico.
4. Trazabilidad:
Grado en el que los datos tienen atributos que proporcionan un camino de acceso auditado a los datos o cualquier otro cambio realizado sobre los datos en un contexto de uso específico.
5. Exactitud:
Grado en el que los datos representan correctamente el verdadero valor del atributo deseado de un concepto o evento en un contexto de uso específico.
Accesibilidad de la información
6. Accesibilidad:
Grado en el que los datos pueden ser accedidos en un contexto específico, particularmente por personas que

Encuesta de usabilidad

Encuesta de usabilidad para la Aplicación web de validación de la norma ISO 25012

El objetivo es medir la interacción de la aplicación web sobre la validación de características y sub-características de la norma ISO 25012 sobre calidad de datos.

Link de la aplicación web se encuentra a continuación: https://evaluacioniso25012.netlify.app/

...

* Obligatorio

1. *

	Totalmente en desacuerdo	En desacuerdo	Ni de acuerdo ni en desacuerdo	De acuerdo	Totalmente de acuerdo
Cree que le gustaría utilizar este sistema con frecuencia	○	○	○	○	○
Encontro el sitio web sensillo	○	○	○	○	○
Piensa que el sitio web es fácil de usar	○	○	○	○	○
Piensa que puede utilizar el sistema sin el apoyo de una persona técnica	○	○	○	○	○
Encontró que varias de las funciones en el sitio web estaban bien integradas	○	○	○	○	○
Piensa que había demasiada inconsistencia en el sistema	○	○	○	○	○
Imagina que la mayoría de la gente aprendería a usar este sistema de forma muy rápida	○	○	○	○	○
Encuentra que el sistema es muy difícil de usar	○	○	○	○	○
Se sintió confiado al usar este sistema	○	○	○	○	○
Puede utilizar el sitio web sin tener que aprender algo nuevo	○	○	○	○	○