

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS.

CARRERA DE SOFTWARE.



DESARROLLO DE UNA GUÍA METODOLÓGICA PARA LA GESTIÓN DE PROYECTOS DE SOFTWARE CON LA HERRAMIENTA GITHUB Y EL MARCO DE TRABAJO SCRUM.

Trabajo de Grado previo a la obtención del título de Ingeniero en Software.

Autor:

Wellington Israel Rochina Rea

Director:

PhD. José Antonio Quiña Mera MSc.

Ibarra, 2023



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:	0202224259		
APELLIDOS Y NOMBRES:	ROCHINA REA WELLINGTON ISRAEL		
DIRECCIÓN:	IBARRA, OLIVOS AV. 17 DE JULIO Y DR. LUIS MADERA		
EMAIL:	wirochinar@utn.edu.ec		
TELÉFONO FIJO:	033033287	TELÉFONO MÓVIL:	0985938790

DATOS DE LA OBRA	
TÍTULO:	DESARROLLO DE UNA GUÍA METODOLÓGICA PARA LA GESTIÓN DE PROYECTOS DE SOFTWARE CON LA HERRAMIENTA GITHUB Y EL MARCO DE TRABAJO SCRUM.
AUTOR:	ROCHINA REA WELLINGTON ISRAEL
FECHA:	20/07/2023
PROGRAMA:	<input checked="" type="checkbox"/> PREGRADO <input type="checkbox"/> POSGRADO
TITULO POR EL QUE OPTA:	INGENIERO DE SOFTWARE
DIRECTOR:	PHD. ANTONIO QUIÑA, MSC.
ASESOR 1:	ING. MAURICIO REA, MSC.
ASESOR 2:	PHD. IRVING REASCOS, MSC.

2. CONSTANCIA

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 20 días del mes de julio de 2023

EL AUTOR:



ESTUDIANTE

Rochina Rea Wellington Israel

C.I: 020222425-9

CERTIFICACIÓN DIRECTOR

CERTIFICACIÓN DIRECTOR

Ibarra 20 de Julio del 2023

CERTIFICACIÓN DIRECTOR DEL TRABAJO DE TITULACIÓN

Por medio del presente yo PhD. Antonio Quiña, MSC, certifico que el Sr. Rochina Rea Wellington Israel portador de la cédula de la ciudadanía número 0202224259, ha trabajado en el desarrollo del proyecto de grado **"DESARROLLO DE UNA GUÍA METODOLÓGICA PARA LA GESTIÓN DE PROYECTOS DE SOFTWARE CON LA HERRAMIENTA GITHUB Y EL MARCO DE TRABAJO SCRUM"**, previo a la obtención del Título de Ingeniero en Software realizado con interés profesional y responsabilidad que certifico con honor de verdad.

Es todo cuanto puedo certificar en honor a la verdad.

Atentamente



PhD. Antonio Quiña, MSc.

DIRECTOR DE TESIS

DEDICATORIA

El presente trabajo lo dedico a mis padres Rea Mercedes y Rochina José, por haber estado presente en el transcurso de mi educación, por todo el esfuerzo y amor que me han depositado dándome seguridad. Espero poder retribuir todo lo han hecho por mí.

A mis hermanos que desde nuestra infancia hemos compartidos secretos, sueños, risas y lágrimas. Sobre todo, han sido mi apoyo incondicional en los momentos más difíciles, quienes me han brindado fuerzas para culminar esta etapa de mi vida.

AGRADECIMIENTO

Agradezco a los docentes de la carrera de software por su sabiduría y apoyo profesional y personal durante mi formación académica en la carrera de software. Ha sido gratificante aprender de ellos a lo largo de mi educación.

También agradezco a mi tutor de trabajo de titulación PhD. Antonio Quiña, MSc. y asesores Ing. Mauricio Rea y PhD. Irvin Reascos, MSc. quienes han compartido sus conocimientos técnicos y experiencia para culminar con mi carrera universitaria.

Ing. Cathy Guevara MSc. Gracias por su ayuda y apoyo en mi primer acercamiento al campo laboral. Sus conocimientos han sido significativos en mi vida profesional.

Gracias a mis amigos por los momentos compartidos de risas, que hicieron días alegres en esta etapa de mi vida.

ÍNDICE DE CONTENIDO

CERTIFICACIÓN DIRECTOR	IV
DEDICATORIA	V
AGRADECIMIENTO	VI
Resumen	XIV
Abstract	XV
INTRODUCCIÓN	1
Antecedentes:	1
Situación actual:	1
Planteamiento del Problema:	2
Objetivos	3
Objetivo General	3
Objetivos Específicos	3
Alcance	3
Justificación	4
Justificación metodológica	4
Justificación técnica	5
CAPÍTULO I	6
1. Marco teórico	6
1.1. Gestión de proyectos de software	6
1.1.1. ¿Qué es un proyecto?	6
1.1.2. ¿Qué es Software?	7
1.1.3. ¿Qué es la gestión de proyectos de software?	7
1.1.3. Determinación del éxito en la gestión de proyectos de software	9
1.1.4. Resultados de éxito en Gestión de Proyectos de Software	12
1.2. Marco de trabajo Scrum	12
1.2.1 Bases teóricas (Manifiesto ágil)	12
1.2.2. Orígenes del marco de gestión SCRUM	14

1.2.3. Teoría de Scrum.....	16
1.2.4. Miembros del equipo SCRUM	18
1.2.5. Eventos en SCRUM.....	20
1.2.6 Artefactos de Scrum	22
1.3. Herramienta GitHub	25
1.3.1. Origen de GitHub “Git”	25
1.3.2. ¿Qué es GitHub?.....	27
1.3.3. Estado de GitHub en la actualidad.....	28
1.3.4. ISO/ IEC 25022	29
CAPÍTULO II	31
2. Diseño de la propuesta.....	31
2.1. Metodología para desarrollar la guía Metodológica.....	31
2.1.1. Definición de la Guía Metodología.....	31
2.1.2. Fases de la Guía Metodológica	31
2.1.3. Diseño de la Guía metodológica	35
2.1.4. Artefactos y Eventos de Scrum, correspondencia con las funcionalidades de GitHub.....	36
2.1.5. Pre-Juego.....	36
2.1.6. Juego	41
2.1.7. Post-Juego	46
CAPÍTULO III	50
3. Desarrollo de la Guía metodológica.....	50
3.1. Introducción	50
3.2 Objetivo.....	50
3.3. Alcance	50
3.4 Audiencia	50
3.5. Ejecución de las fases Scrum	51
3.5.1. Fase 1. Pre-Juego	51
3.5.2. Fase 2. Juego.....	58

3.5.3 Fase 3. Post-Juego	76
CAPÍTULO IV	79
4. Validación de resultados.....	79
4.1. Desarrollo de la prueba de concepto.....	79
4.1.1. Fase 1. Pre-Juego	79
4.1.2 Fase 2. Juego.....	82
4.1.3. Fase 3. Post-Juego	92
5.1 Resultados de la prueba de concepto	99
5.1.1. Análisis de resultados.....	99
5.2.1 Evaluación de los resultados	101
Conclusiones.....	103
Recomendaciones.....	104
Referencias	105
ANEXOS	109

ÍNDICE DE FIGURAS

Figura 1	Diagrama de problemas.....	2
Figura 2	Arquitectura de la prueba de concepto.	4
Figura 3	Gestión de proyecto.....	8
Figura 4	Estimación de costo.....	9
Figura 5	Valores del manifiesto ágil.	13
Figura 6	Valores en la metodología ágil.....	14
Figura 7	Roles o miembros de scrum.	18
Figura 8	Fases de spring.	22
Figura 9	Fases de scrum	24
Figura 10	Almacenamiento de etapas en GitHub.....	26
Figura 11	Sistemas de control de versiones distribuidas.	27
Figura 12	Arquitectura GitHub.	28
Figura 13	Fases para desarrollar una guía metodológica	32
Figura 14	Fase 1 Pre-Juego	37
Figura 15	Collaborators (Rol).....	39
Figura 16	Issues	40
Figura 17	Labels.....	41
Figura 18	Fase 2 Juego.....	42
Figura 19	Iteraciones.....	44
Figura 20	Milestone	44
Figura 21	Assignees.....	45
Figura 22	Projects.....	45
Figura 23	Fase 3 Post-Juego.....	46
Figura 24	Tags	48
Figura 25	Release	49
Figura 26	Creación del Repositorio GitHub.....	52
Figura 27	Comandos para inicializar el repositorio GitHub.....	53
Figura 28	Agregar colaboradores al proyecto.	54
Figura 29	Creación de Issue.....	56
Figura 30	Añadir tarea a la issue.	56
Figura 31	Completar una actividad en la Issue.	57
Figura 32	Crear Label.....	58
Figura 33	Creación de discusión.....	59
Figura 34	Activar discusión.....	59
Figura 35	Creación de milestone.	60
Figura 36	Asignar Issue a Milestone.....	61

Figura 37	Lista de issues en milestone.....	61
Figura 38	Ingresar al perfil para crear un proyecto.....	62
Figura 39	Seleccionar la plantilla para el proyecto.....	62
Figura 40	Configurar el proyecto.....	62
Figura 41	Opciones de configuración del proyecto	63
Figura 42	Configurar iteraciones en el proyecto.....	64
Figura 43	Añadir más iteraciones.	65
Figura 44	Agregar Issue.	65
Figura 45	Asignación de iteraciones a las issues.....	66
Figura 46	Estados en el proyecto.	66
Figura 47	Asignar tarea.	67
Figura 48	Asignar responsables de las issues desde el Project.....	68
Figura 49	Crear una rama por un Product Backlog.....	69
Figura 50	Configuración para crear una rama nueva.....	69
Figura 51	Sprint Daily Scrum.....	70
Figura 52	Creación Issue Daily Scrum.....	70
Figura 53	Daily scrum en la iteración.....	70
Figura 54	Creación milestone Sprint Review	71
Figura 55	Asignar sprint review a la iteración (Sprint).....	71
Figura 56	Close Issue.....	73
Figura 57	Finalizar issue en la iteración.....	73
Figura 58	Realizar commit para cerrar una Issue.....	74
Figura 59	Compare & pull request.	74
Figura 60	Create pull request.	74
Figura 61	Merge Pull Request	75
Figura 62	Confirm Merge.....	75
Figura 63	Eliminar una rama de Issue.	75
Figura 64	Creación Sprint Retrospective.	76
Figura 65	Asignación sprint retrospective a iteración.....	76
Figura 66	Crear Tag.	77
Figura 67	Subir Tag al repositorio Remoto.....	77
Figura 68	Tag Creado y subido al repositorio GitHub.	77
Figura 69	Acceder a crear Releases.....	78
Figura 70	Creación de Releases.....	78
Figura 71	Inicialización del repositorio.	79
Figura 72	Comandos para inicializar el repositorio.....	80
Figura 73	Archivos cargados desde la maquina local.	80

Figura 74 Etiquetas de nivel dificultad y Prioridad prueba de concepto.....	81
Figura 75 Product Backlog de la prueba de concepto.....	82
Figura 76 Discusión de la estructura de los Sprint.	82
Figura 77 Creación de hitos prueba de concepto.	83
Figura 78 Sprint Backlog de la prueba de concepto.....	83
Figura 79 Iteraciones creadas para la prueba de concepto.....	84
Figura 80 Sprint 1 de la prueba de concepto.	84
Figura 81 Sprint 2 de la prueba de concepto.	85
Figura 82 Sprint 3 de la prueba de concepto.	86
Figura 83 Sprint 4 de la prueba de concepto.	86
Figura 84 Asignación de tareas prueba de concepto	87
Figura 85 Realización del sprint 1 prueba de concepto.....	87
Figura 86 Realización del sprint 2 prueba de concepto.....	88
Figura 87 Realización del sprint 3 prueba de concepto.....	88
Figura 88 Realización del sprint 4 prueba de concepto.....	89
Figura 89 Daily scrum prueba de concepto.....	89
Figura 90 Daily scrum sprint 1 prueba de concepto	89
Figura 91 Daily scrum sprint 2 prueba de concepto.	90
Figura 92 Daily scrum sprint 3 prueba de concepto.	90
Figura 93 Daily scrum sprint 4 prueba de concepto.	90
Figura 94 Sprint review prueba de concepto.....	91
Figura 95 Sprint review asignado a las iteraciones prueba de concepto.	91
Figura 96 Sprint retrospective prueba de concepto.....	91
Figura 97 Sprint retrospective asignado a las iteraciones prueba de concepto.	92
Figura 98 Tags de la prueba de concepto.....	92
Figura 99 Entregable final de la prueba de concepto.	93
Figura 100 Diagrama de BD prueba de concepto.	94
Figura 101 Repositorio backend prueba concepto.	95
Figura 102 Repositorio frontend prueba concepto	95
Figura 103 Arquitectura backend prueba de concepto.....	96
Figura 104 Arquitectura frontend prueba de concepto	96
Figura 105 Gestión de proyecto prueba de concepto.....	97
Figura 106 Servicio rest prueba de concepto.....	98
Figura 107 Página web inicio catálogo vehículos.....	98
Figura 108 Página web más información vehículo.	99
Figura 109 Niveles de calidad.....	102

ÍNDICE DE CUADROS

Tabla 1 Definición de proyecto.....	6
Tabla 2 Factores que influyen en el éxito de un proyecto.....	10
Tabla 3 Variables que influyen en el desempeño de un proyecto.....	11
Tabla 4 Premisas de Scrum.....	15
Tabla 5 Definición de transparencia.....	17
Tabla 6 Definición de Inspección.....	17
Tabla 7 Definición de adaptación.....	18
Tabla 8 Medidas de efectividad.....	29
Tabla 9 Artefactos y eventos de Scrum con las funcionalidades de GitHub.....	36
Tabla 10 Estructura del Product Backlog.....	54
Tabla 11 Estructura de Sprint Backlog.....	64
Tabla 12 Roles dentro del proyecto.....	81
Tabla 13 Evaluación de la guía mediante la completitud funcional.....	100

Resumen

El desarrollo de software se ha visto fuertemente impulsado en el mundo empresarial provocado por los beneficios y la digitalización de las empresas para gestionar sus procesos. Sin embargo, muchos proyectos de software han fracasado por falta de gestión y el uso correcto de las herramientas tecnológicas como GitHub y el marco de trabajo SCRUM. Los profesionales y estudiantes no tienen suficiente conocimiento para aprovechar estas herramientas, los pocos proyectos que han llegado a tener éxito han sido por la experiencia de los encargados por tal motivo es necesario desarrollar un guía.

Este proyecto está enfocado en desarrollar una guía metodología apoyándose de dos herramientas que están acentuados en el mundo de software "GitHub y SCRUM". Se llevo a cabo una investigación de las fases y funcionalidades que tiene estas dos herramientas, tras esto se diseñó la guía metodológica dividido en tres fases de SCRUM (Pre-Jugo, Juego, Post-Juego) las mismas que hacen correlación con las funciones de GitHub (Issue, Projects, Milestone, Branch, Tag y Release)

La guía fue validada mediante una prueba de concepto, se realizó un aplicativo web haciendo uso de la guía. Se identifico factores que influyen en la implementación de la guía. La guía fue evaluada con la ISO/IEC 25022 "Calidad en uso", los resultados demostraron que la guía tiene un grado de satisfacción oportuno.

La guía final obtenida, ayuda a gestionar proyectos de software aumentando el grado de éxito. La guía detalla con ilustraciones los pasos a seguir.

Palabras claves: GitHub, Scrum, gestión, proyectos y software.

Abstract

The development of software has been strongly driven in the business world, spurred by the benefits and digitalization of companies to manage their processes. However, many software projects have failed due to lack of management and proper use of technological tools such as GitHub and the SCRUM framework. Professionals and students lack sufficient knowledge to leverage these tools, and the few projects that have succeeded have done so due to the experience of those in charge. Therefore, it is necessary to develop a guide.

This project focuses on developing a methodology guide, leveraging two tools that are prominent in the software world: "GitHub and SCRUM." An investigation was conducted on the phases and functionalities of these two tools, and based on this, the methodological guide was designed, divided into three SCRUM phases (Pre-Game, Game, Post-Game) that correlate with the functions of GitHub (Issue, Projects, Milestone, Branch, Tag, and Release).

The guide was validated through a proof of concept, where a web application was developed using the guide. Factors influencing the implementation of the guide were identified. The guide was evaluated using ISO/IEC 25022 "Quality in Use," and the results demonstrated that the guide achieved a satisfactory level of satisfaction.

The final guide obtained helps manage software projects and increases the degree of success. The guide provides detailed steps with illustrations to follow.

Keywords: GitHub, Scrum, management, project and software.

INTRODUCCIÓN

Antecedentes:

El desarrollo de software va en crecimiento exponencialmente, en donde las industrias y la sociedad tienen nuevas necesidades y requieren de software que automaticen sus procesos (Espinoza Mina & Gallegos Barzola, 2017). Por lo cual es necesario que se lleve a cabo una buena gestión del desarrollo de software, tras estas necesidades se han creado herramientas que ayuden en la gestión de procesos.

A pesar de contar con herramientas para la gestión de proyectos, no son utilizadas de la manera más eficiente, este es el caso de la herramienta GitHub, no ha tenido una curva de aprendizaje exitosa, las personas que usan GitHub desconocen de cómo emplearla, no existe un lineamiento de su uso.

De acuerdo con el reporte de Standish Group, clasifica a los proyectos en exitosos, problemáticos, deficientes sus porcentajes correspondientes son 6.2%, 56%, 15% (Romero-Romero & López-Botello, 2020).

Los proyectos de software tienen graves problemas en soporte ejecutivos, falta de gestión, provocando un alto costo de tiempo y recursos en el desarrollo de software.

Situación actual:

Actualmente en la industria de software, en la carrera de software y sistemas computacionales de la Universidad técnica del Norte (UTN), durante el desarrollo de un software no se gestiona de manera correcta o formal, se toma poco interés en el proceso del desarrollo del software, existe falta de diagnóstico de proyectos, llega a tal punto de que tratan de reducir los costos de producción.

GitHub es una herramienta que se puede utilizar en el uso de aprendizaje colaborativo (Zakiah & Fauzan, 2016). Sin embargo, los proyectos que se han realizado con GitHub tienen deficiencia en la gestión del desarrollo de software, las gestiones que se realizan son realizadas por experiencias propias de cada individuo, esto no garantiza el éxito del proyecto. Desconocen del manejo de GitHub, esto genera retrasos, incidentes, incertidumbres alejándose del objetivo central de un proyecto de software.

Otro inconveniente en el desarrollo de software es que existe falta de organización en los grupos de trabajo, mala planificación, falta de asignación de responsabilidades, llegan a sobrestimar el tiempo y recursos para el desarrollo de software (López et al., 2017).

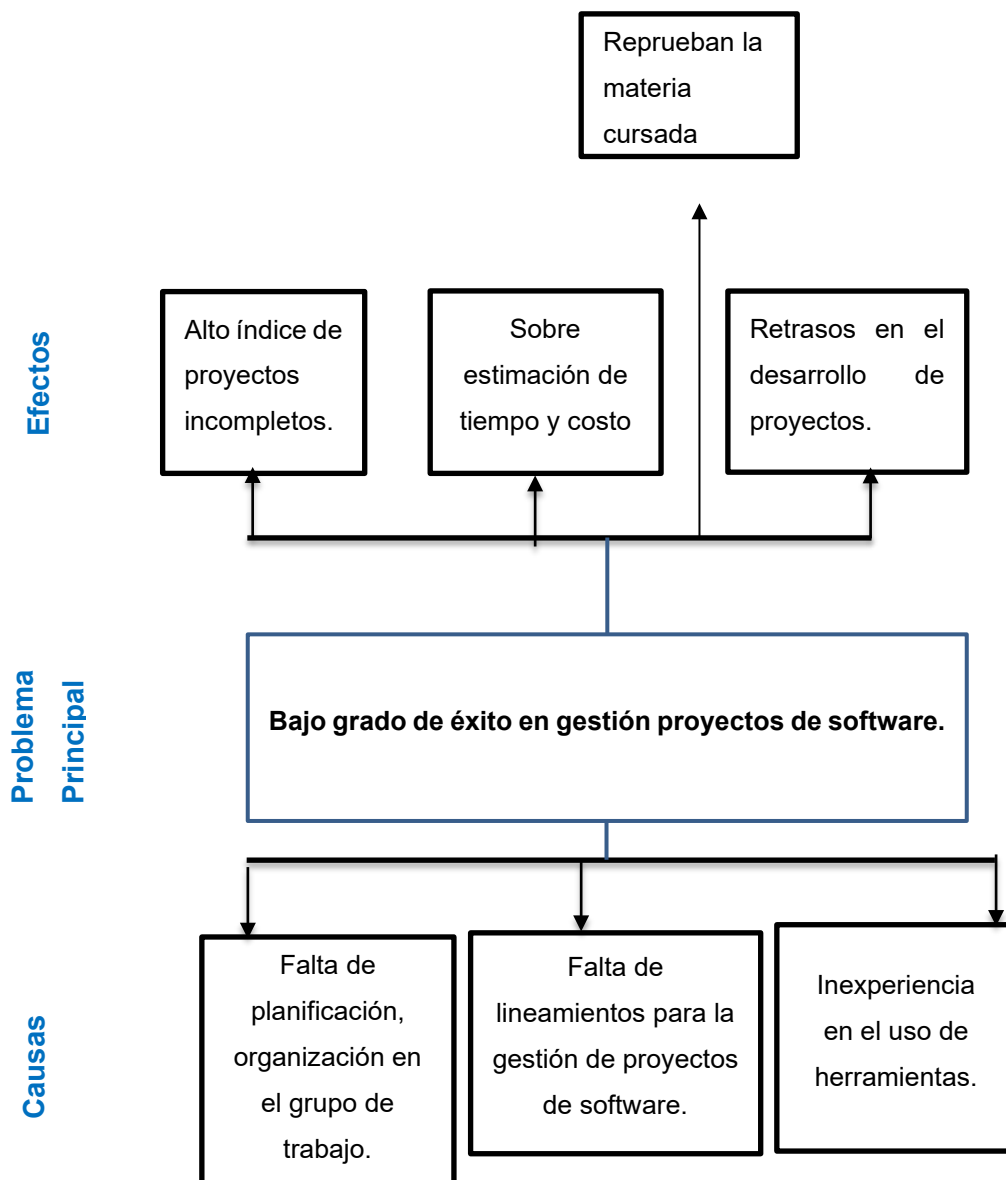
Planteamiento del Problema:

Existe bajo grado de éxito en la gestión de proyectos de software por parte de los docentes y estudiantes de la carrera de ingeniería en sistemas y software, el motivo principal es la ausencia de una gestión adecuada, falta de entendimiento y aplicación de un marco de trabajo que permita aumentar el grado de éxito (Kuz et al., 2018).

El uso de la herramienta GitHub es otro factor que aumenta la calidad del desarrollo de un software, también ayuda a los estudiantes en la gestión de proyectos (Kertesz, 2015). Sin embargo, los estudiantes de la carrera de software y sistemas computacionales de la UTN tienen poco conocimiento del manejo de la herramienta GitHub u otro gestor de proyectos, esto provoca que la gestión de proyectos de software sea deficiente.

Figura 1

Diagrama de problemas



Objetivos

Objetivo General

Desarrollar una Guía metodológica para la gestión de proyectos de software con la herramienta GitHub y el marco de trabajo Scrum.

Objetivos Específicos

- Establecer marco teórico y tecnológico del uso de GitHub y Scrum.
- Desarrollar una guía metodológica de gestión de proyecto integrando GitHub y Scrum.
- Validar la guía metodológica mediante una prueba de concepto basado en la norma ISO/IEC 25022 medición de calidad en uso.

Alcance

El presente trabajo tiene como objetivo elaborar una guía metodológica para la gestión de proyectos de software con la herramienta GitHub orientándose al marco de trabajo Scrum. Esto permitirá fortalecer la gestión de proyectos de desarrollo de software para minimizar el tiempo y costo de desarrollo. La guía va enfocada a los estudiantes de las carreras de software y sistemas computacionales de la Universidad Técnica del Norte.

Para cumplir con los objetivos planteados en este proyecto se ha dividido el trabajo en las siguientes etapas:

Etapas 1: para cumplir con el objetivo 1, se establecerá un marco teórico y tecnológico; mediante una investigación bibliográfica para identificar las principales características de GitHub, los principios de Scrum, ¿Quiénes se involucran? Además, se realizará una revisión de la literatura para identificar trabajos similares al propuesto en esta investigación.

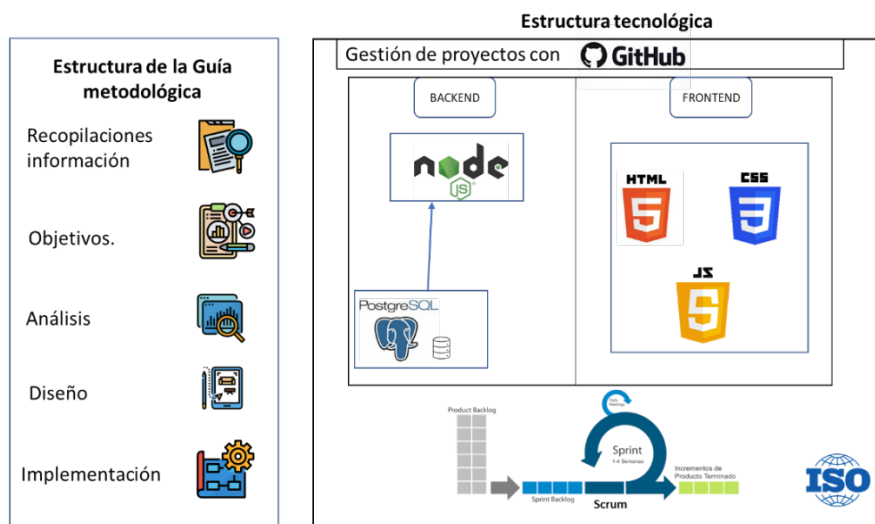
Etapas 2: Para cumplir con el objetivo 2, se procederá a elaborar la guía de gestión de proyectos, se buscará la manera de integrar GitHub y el marco de trabajo Scrum, cada principio de Scrum se integrará con una de las funcionalidades de GitHub para gestionar proyectos de desarrollo de software.

Etapas 3: Para cumplir con el objetivo 3, se medirá la validación de la guía usando las métricas de la característica de eficacia basado en la norma ISO/IEC 25022 (calidad en uso), esto permite satisfacer los criterios de calidad (Reina Guña et al., 2019). La validación consiste en usar la guía metodológica en el desarrollo de una prueba de concepto.

La prueba de concepto se refiere al desarrollo de una aplicación web de catálogo de ventas de carros (básico). Las herramientas tecnológicas para utilizar son: base de datos PostgreSQL, en la parte de backend con NodeJs, como frontend JavaScript, Css, Html.

Figura 2

Arquitectura de la prueba de concepto.



Justificación

Justificación metodológica.

La falta de lineamientos para la integración de Scrum y GitHub en el desarrollo de un software, gestión de proyectos ejecutados de manera ineficiente, la inexperticia de planificar y trabajar en equipo, junto a esto que el progreso del proyecto tenga una trazabilidad y sea entregado los proyectos en los cronogramas estimados hace necesario el desarrollo de una guía para la gestión de proyectos de software.

En cuanto a los objetivos del ODS, se contemplan.

Objetivo 4 Educación de calidad: Al concluir con la realización de la guía se pretende contribuir el acceso al conocimiento contenido en este proyecto (ODS, 2022).

Objetivo 9 Industria innovación e infraestructura: Dar acceso a la tecnología de información de manera universal y asequible desde internet. Apoyar al desarrollo tecnológico e investigaciones (ODS, 2022).

Justificación técnica.

Desarrollar una guía metodológica para la gestión de proyectos de software que integre la herramienta GitHub con el marco de trabajo Scrum, apoyada del estándar ISO/IEC 25022. Esto ayudará en la gestión de proyectos a los estudiantes de la carrera de software y sistemas computacionales de la UTN.

CAPÍTULO I

1. Marco teórico

Este capítulo establece el marco conceptual y tecnológico para realizar el desarrollo de una guía enfocada a la gestión de proyecto de software utilizando GitHub, marco de trabajo Scrum. Esta sección está dividida en: ¿Qué es un proyecto?, ¿Qué es software?, ¿Qué es la gestión de proyectos de software?, Determinación de éxito en la gestión de proyectos de software, Resultados de éxito en la gestión de proyectos de software.

1.1. Gestión de proyectos de software

1.1.1. ¿Qué es un proyecto?

Se define a un proyecto como el interés propuesto para cumplir o querer satisfacer las necesidades de la sociedad, estas necesidades se convierten en productos o bienes. Dentro del interés propuesto se involucran el tiempo, entusiasmo y esfuerzo aplicado para lograr cumplir con un objetivo. Aparte existen otras definiciones como las que se observan en la Tabla 1.

Tabla 1

Definición de proyecto.

Autores/estándares	Definición
Managing successful Project with PRINCE2	Es una organización temporal que se crea con el propósito de entregar uno o más productos comerciales, pueden ser bienes o servicios de acuerdo con un caso comercial acordado (The Stationery Office & Lea, 2017).
Gestión de proyectos aplicada al PMBOK 6ED	Es un esfuerzo especial que se aplica para obtener un resultado; producto o servicio esperando que satisfaga las necesidades o requisitos de un grupo de interesados (Sarmiento Rojas et al., 2020).
PMBOK	Es un esfuerzo temporal que lleva a cabo para crear un producto, servicio o resultado único (Guía del PMBOK®, 2008).

1.1.2. ¿Qué es Software?

Entendemos como software aquel componente lógico que es intangible, no físico. Realiza una tarea en específica en algún sistema. Un software es un intermediario entre los componentes físicos de la computadora y la persona. Da ordenes al hardware (componente tangible, físico) encomendadas por el usuario. El software se define como el conjunto de procedimientos, documentación, programas de cómputo que son parte de las operaciones del sistema (Moreno Perez, 2015).

El software está formado por un conjunto de reglas para realizar trabajos en las computadoras, brinda soluciones a las necesidades del usuario. Se construye a medida, no se fabrica se desarrolla, es intangible. El software requiere de soporte, consultoría y capacitación (Rodriguez Moreno, 2020).

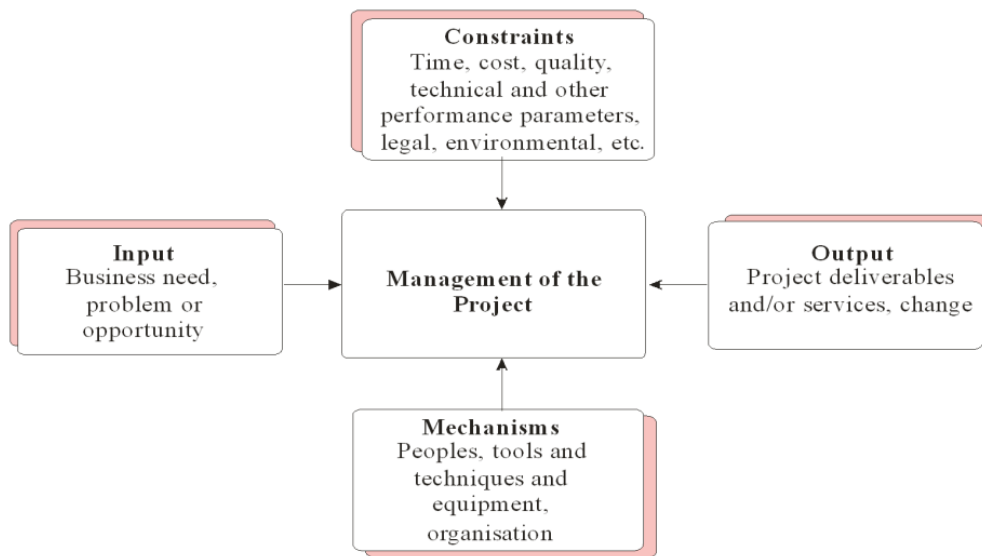
1.1.3. ¿Qué es la gestión de proyectos de software?

La gestión de proyectos es la planificación, delegación, monitoreo y control de todos los aspectos de un proyecto, y la motivación de los involucrados para lograr los objetivos del proyecto dentro de los objetivos del desempeño en cuanto a tiempo, costo, alcance, beneficios, riesgos y calidad (The Stationery Office & Lea, 2017).

La Gestión de Proyectos de Software implica demandas contrapuestas con respecto al alcance, el costo, el riesgo y la calidad, en ocasiones se involucran a partes interesadas con diferentes necesidades, expectativas y beneficios. Impactan en el logro de los objetivos mutuamente acordados de cualquier proyecto requiere la aplicación de conocimientos especializados, habilidades, herramientas y técnicas a las actividades del proyecto (Diugwu et al., 2015).

Figura 3

Gestión de proyecto.



Nota. Fuente (Diugwu et al., 2015).

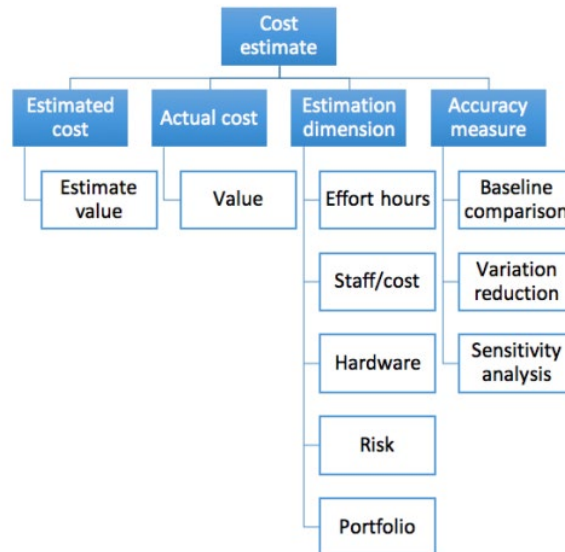
Una gestión de proyectos de software es un proceso de planear, organizar, controlar, monitorear y liderar un proyecto de software. Es importante coordinar varias actividades con múltiples actores o unidades organizacionales con el fin de reducir los costos y tiempo de desarrollo sin dejar de lado a la calidad de software (El Bajta et al., 2018)

Dentro de gestión de proyectos de software existen un factor importante que puede afectar la calidad del producto, se trata de la estimación del costo, consiste en los factores tales como:

- Alcance del proyecto
- Los requisitos del producto
- Capacidad del equipo frente al proyecto

Las principales facetas para estimar el costo según (El Bajta & Idri, 2019) son: **Costo Estimado** en esta faceta se documenta el costo estimado a merced del experto. **Costo Real** es importante tener el costo real al final de la planificación, para poder comparar con el costo estimado. **Dimensión de la estimación** en esta faceta se documenta el esfuerzo de desarrollo como horas de esfuerzo total. **Medida de precisión** permite evaluar el desempeño de la técnica usado para estimar costos como se observa en la *Figura 4*.

Figura 4
Estimación de costo.



Nota. Fuente (El Bajta & Idri, 2019)

1.1.3. Determinación del éxito en la gestión de proyectos de software

El éxito de un proyecto es ambiguo, se puede evaluar desde diferentes puntos de vistas, no existe un método único para medir el éxito de un proyecto, está determinado por diferentes factores (Lamprou & Vagona, 2018). El éxito tanto como la determinación de un logro es ampliamente discutido en la literatura está determinado entre varias partes interesadas (Toor & Ogunlana, 2010).

Las definiciones del éxito del proyecto son ambiguas debido a que el éxito del proyecto se da desde dos puntos de vistas diferentes a nivel macro y micro, dentro del punto macro en la mayoría de los casos se encuentra usuarios finales y los beneficiarios del proyecto. Por otro lado, desde el punto micro hace referencia a la parte de la construcción están los contratistas y consultores por lo general toman como eje principal el tiempo para medir el éxito (Toor & Ogunlana, 2010).

Desde otra perspectiva el éxito gira entorno a cuatro dimensiones (dimensiones: comunicación tiempo, satisfacción de las partes interesadas y costo/presupuesto), las dimensiones se encuentran presentes dentro de las organizaciones (Cliente, ejecutivo, patrocinador, propietario, usuario, equipo del proyecto) (Davis, 2017).

Según (Business Review, 2017) menciona las cuatro etapas esenciales para que una gestión de proyectos de software se considere exitoso: Planificación Desarrollo, Ejecución y

Finalización. También describe las variables que suelen determinar el éxito, **Calidad** = tiempo + coste. Siempre se debe buscar un equilibrio entre estas variables.

Para obtener el éxito de un proyecto, el grupo de trabajo debe: Seleccionar un proceso adecuado para cumplir con los requerimientos de los interesados, tener un enfoque definido que permita lograr el cumplimiento de los requisitos, con el fin de compensar las necesidades de los interesados y lo más importante equilibrar el alcance, tiempo, costo, recursos, riesgos y calidad para obtener un servicio o producto de calidad (Guía del PMBOK®, 2008).

En la determinación del éxito de un proyecto se involucra varios factores que afecta directa o indirectamente al proyecto estos son madurez emocional, participación del usuario, optimización, patrocinio de ejecutores, recursos calificados, arquitecturas estandarizadas, implementación de metodologías ágiles, ejecución modesta, capacidad de gestión de proyectos y capacidad empresarial. En la Tabla 2 se contempla el porcentaje de influencia en cada factor.

Tabla 2

Factores que influyen en el éxito de un proyecto.

Factores influyentes	Porcentaje de influencia
Madurez emocional	15%
Participación del usuario	15%
Optimización	15%
Patrocino de ejecutores	15%
Recursos Calificados	10%
Arquitecturas estandarizadas	8%
Implementación de metodologías ágiles	7%
Ejecución modesta	6%
Capacidad de gestión de proyectos	5%
Capacidad empresarial	4%

Nota. Fuente (Shafiq et al., 2018).

En el desarrollo de un proyecto de software también existe variables que afectan el desempeño entre estos están el costo, plazos, calidad, alcance, beneficio, riesgo. Existe consideraciones a tener en mente dentro de estas variables como se contempla en la Tabla 3.

Tabla 3

Variables que influyen en el desempeño de un proyecto.

Variable	Consideraciones
Costo	El proyecto debe ser asequible, aunque empecemos con un presupuesto en mente surgen muchos factores que pueden conducir a gastos excesivos, también puede haber oportunidades para reducir el presupuesto.
Plazos	Están relaciones con los costos, fecha para la entrega de un proyecto.
Calidad	Los productos deben ser aptos para el propósito, el resultado del proyecto debe de funcionar.
Alcance	El alcance del proyecto debe se bien definido entre el jefe del proyecto y el cliente. No se pueden pasar del alcance esto puede ser motivo de retraso o gastos excesivos.
Beneficio	El proyecto debe ser consistente con el logro del rendimiento deseado.
Riesgo	Siempre existe un riesgo en proyectos, la pregunta es ¿Qué estamos dispuestos hacer para mitigar los riesgos?

Nota. Fuente (The Stationery Office & Lea, 2017).

1.1.4. Resultados de éxito en Gestión de Proyectos de Software

La industria de software tiene un crecimiento exponencial, las empresas que desarrollan software tienen competencia de calidad y de satisfacción del producto hacia el cliente. Standish Group es una organización enfocada en analizar los éxitos y fracasos de los proyectos. Tras un análisis Standish Group emitió un informe donde se muestra los siguientes resultados: el **19%** catalogados como fracasos, el **29%** fueron exitosos, **52%** como discutidos (no se saben si fracasaron o tuvieron éxito) las investigaciones se realizaron en el 2015 (López et al., 2017).

La documentación es un rol fundamental cuando se realizar proyectos pequeños, medianos y grandes en el área de software. En un estudio realizado por (Terlizzi et al., 2016), se determinó que 85% no documentan los requerimientos, el 97% no registran los riesgos.

En la encuesta realizado por (KPMG, 2013), identificaron que la construcción de proyectos va en aumento al igual que la tasa de fracaso, el 33% de proyectos se culminó dentro del presupuesto planificado, el 29% se realizó dentro del tiempo establecido y el 35% por cumplir. Fue menor a la encuesta realizada en el 2010, el 48% se encontraba dentro del presupuesto, el 36% a tiempo y el 59 % por cumplir.

Después de un estudio realizado a 3047 proyectos por (Terlizzi et al., 2016), se determinó las 5 principales barreras para el uso de las metodologías de gestión de proyectos (**MGP**) son:

- Los plazos de los proyectos son muy ajustados
- Trabajar como gerente y desarrollador de proyecto
- Trabajar simultáneamente en varios proyectos
- Dificultad en el uso de MGP
- Poco conocimiento de las MGP

1.2. Marco de trabajo Scrum

1.2.1 Bases teóricas (Manifiesto ágil).

El marco de trabajo Scrum se deriva de la **metodología ágil**. Busca realizar tareas de manera iterativa y con mejoras continuas.

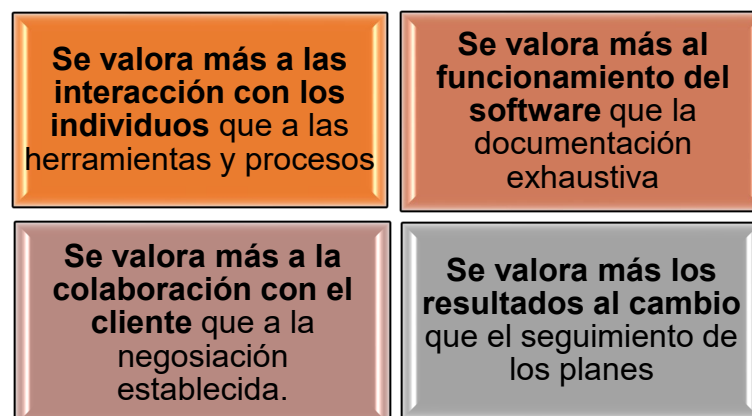
Las metodologías ágiles están enfocadas en el recurso humano y la calidad del producto de software, se centran en dar más valor a los interesados del proyecto de software y a realizar tareas de manera iterativa.

La metodología ágil, se interesa en enfrentas a retos de nivel alto y pretende realizar participaciones continuas con los interesados de un proyecto. Tiene el enfoque de gestionar las actividades de manera flexible, iterativa y crece de manera evolutiva (Mondelo & Sanchez Orduña, 2018). Es un modelo que resuelve proyectos con alto grado de incertidumbres, fomenta el trabajo grupal, se caracteriza por estar preparado para los errores para este modelo solo es un medio de aprendizaje.

Dentro del manifiesto ágil existen valores que no pueden ser obviados o ignorados, son considerados núcleo del manifiesto ágil. Son 4 valores, como se observar en la Figura 5.

Figura 5

Valores del manifiesto ágil.



Nota. Fuente (Mondelo & Sanchez Orduña, 2018).

La metodología ágil, está centrado en el cambio de mentalidad, esto no funciona de manera individual debe ser acogido por todos los miembros del equipo que estén dispuesto a realizar cambios en sus maneras de trabajar. Los miembros del equipo deben relacionarse con seis valores que deben estar presentes en el cambio de mentalidad hacia la metodología ágil como se observa en la Figura 6.

Figura 6

Valores en la metodología ágil.



Nota. Fuente (Mondelo & Sanchez Orduña, 2018).

1.2.2. Orígenes del marco de gestión SCRUM

Scrum es un marco de trabajo derivado de la **metodología ágil** y elaborado sobre las bases del empirismo, fue desarrollado en el año 1993, fundado por Ken Schwaber y Jeff Shutherland. Inspirado del juego Rugby, se enfocaron en la estructura holística y organizada del equipo durante el juego (Bhavsar & Gopalan, 2020).

Scrum también se basa en el proceso de mejora continua, esto se detalla en el ciclo de calidad PHVA. Dentro de este ciclo están: Planear, Hacer, Verificar y actuar (Zapata Gomez, 2015).

Planear. Se establecen objetivos a cumplir, se realizan los preparativos para cumplir la lista de metas de calidad, lista de necesidades de los clientes, prototipos, diseños, actividades a realizar antes de poner a producir el producto. El objetivo de la planificación es suministrar las fuerzas para satisfacer las necesidades del cliente. Es importante tener la misión y visión del proyecto.

Hacer. Es la implementación de lo que se ha planificado, identificación de las mejoras al proyecto. Es el proceso en el cual se transforma la planeación, recursos en un producto o servicio.

Verificar. En esta parte implica la revisión de las actividades, verificar que se están cumpliendo con los planes para lograr el objetivo de satisfacer al cliente. Se informa sobre los resultados, se comparan los datos recolectados con la planificación.

Actuar. Busca soluciones para mejorar las próximas planeaciones, centrado en realizar un ciclo de mejora en todos los niveles. Incentiva al grupo de trabajo, soluciona problemas de los miembros del equipo. No termina al tener un resultado satisfactorio, a lo contrario inicia un nuevo desafío.

Scrum busca fortalecer los trabajos colaborativos, anima al equipo aprender mediante experiencias, Scrum contiene herramientas, reuniones y varias funciones que de manera coordinada logran gestionar un proyecto (Morandini et al., 2021). Para cumplir con estos objetivos Scrum plantea premisas que se debe cumplir en el desarrollo del proyecto como se contempla en la Tabla 4.

Tabla 4
Premisas de Scrum.

Premisa	Definición
Satisfacción del cliente	El objetivo es satisfacer al cliente. El cliente debe recibir lo que desea y debe de sentirse satisfecho con el producto entregado. El producto debe ser útil para él.
Receptividad ante el cambio de requerimientos.	Los requisitos que se obtienen no son estáticos pueden cambiar dependiendo de la situación, se debe prever y confrontar este hecho.
Trabajar enfocado en el proyecto, producto o servicio.	El fin es construir un producto útil, por lo que el método empleado está en segundo plano.
Desarrollo Sostenible.	La creación del producto no debe afectar a los involucrados del proyecto.

Cooperación diaria, abierta entre desarrolladores y negocio.	La información no puede estancar, toda la información debe ser clara entre los involucrados del proyecto.
Comunicación directa entre personas.	No debe haber ocultamiento de problemas entre los miembros del proyecto, comunicación de cara a cara.
Individuos motivados vs individuos dirigidos	Los miembros del equipo no deben sentirse excluidos, deben formar parte del equipo. Los miembros deben tener asignados actividades, también deben ser partícipes de las decisiones.
Orientación a la excelencia	Los productos que se crean deben tener calidad, incremento de calidad cada día,
Simplicidad	Cumplir y hacer solo lo que se ha planteado ni más ni menos.
Equipos autoorganizados.	El equipo puede organizar de tal manera que puedan cumplir con los objetivos. No trabajan de manera individual.
Adaptabilidad.	Es necesario adaptarse a los cambios imprevistos dentro del proyecto. La adaptabilidad se logra únicamente si el equipo es adaptable.

Nota. Fuente (Monte Galiano, 2016).

1.2.3. Teoría de Scrum

Scrum está basado en el pensamiento Lean y el empirismo. El empirismo tiene el concepto de que el conocimiento proviene de la experiencia y toma de las decisiones con respecto a lo observado. Mientras el pensamiento Lean se enfoca en la entrega del valor, no faltar el respeto a las personas, minimización de información basura, existencia de la transparencia, adaptación al cambio y existencia de la mejora continua siempre enfocándose a lo esencial (Project Management Institute, 2017).

Scrum busca integrar a personas que tienen experiencia y habilidades de compartir el conocimiento entre los miembros del equipo, Scrum combina 4 eventos para la adaptación e

inspección que lo denomina Sprint. Los eventos funcionan de manera efectiva porque implementan tres pilares fundamentales: adaptación, transparencia e inspección (Schwaber & Sutherland, 2020). Ver Tabla 5, Tabla 6, Tabla 7.

Transparencia.

Tabla 5

Definición de transparencia.

Guía.	Definición
PM4 agile	Las planificaciones y facetas del proyecto deben ser visible para todo los involucrados (Mondelo & Sanchez Orduña, 2018).
The Scrum Guide	El proceso debe ser visible para todos los miembros del equipo como también para quienes lo reciben (Schwaber & Sutherland, 2020).

Inspección.

Tabla 6

Definición de Inspección.

Guía.	Definición
PM4 agile	significa realizar inspecciones cada cierto tiempo para verificar cuán bien están avanzado el logro de los objetivos planificados (Mondelo & Sanchez Orduña, 2018).
The Scrum Guide	Se enfoca en detectar problemas que no se tomaron en cuenta en la planeación, la inspección facilita la adaptación (Schwaber & Sutherland, 2020).

Adaptación.

Tabla 7

Definición de adaptación.

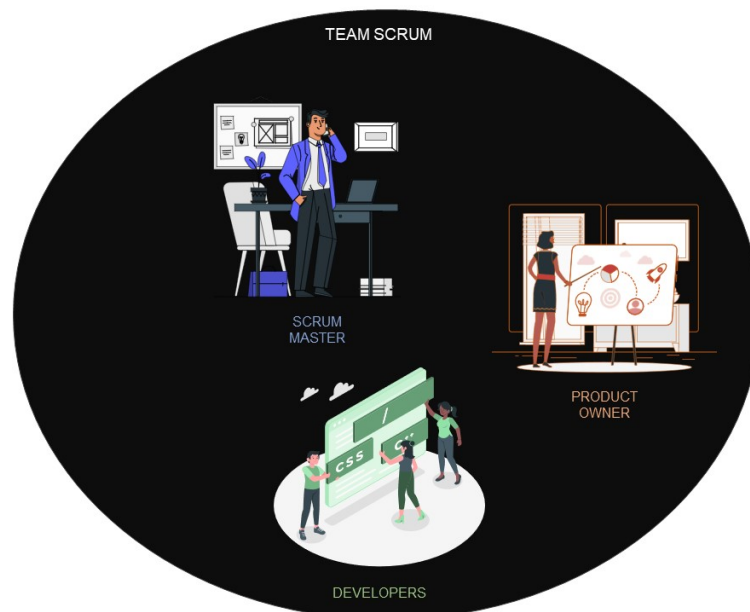
Guía.	Definición
PM4 agile.	Se trata de aprender y luego mejorar para las próximas iteraciones (Mondelo & Sanchez Orduña, 2018).
The Scrum Guide.	Trata de minimizar las desviaciones que se encuentran fuera de los límites aceptables (Schwaber & Sutherland, 2020).

1.2.4. Miembros del equipo SCRUM

La *Figura 7* muestra los miembros del equipo, estos son: Product Owner, Scrum Master y Developers. Tener en cuenta; dentro del grupo no existe jerarquías. Los miembros del equipo se autogestionan por sí mismas es decir que son capaces de tomar decisiones y cumplir con el objetivo de obtener un producto (Schwaber & Sutherland, 2020).

Figura 7

Roles o miembros de scrum.



La cantidad de miembros dentro de un equipo no debe sobrepasar de las 10 personas, es la cantidad suficiente para cumplir con los objetivos y no salir de la metodología ágil. Cada miembro del equipo esta apto para cumplir con los Sprint planteados.

Los equipos se centran en desarrollar los productos de manera rápida y eficaz, siempre esperan una retroalimentación. Los integrantes del equipo siempre buscan lograr los objetivos en equipo no de manera individual. Tienen las habilidades necesarias para entregar los trabajos.

Desarrollador (Developers)

Los desarrolladores son las personas que se proponen a realizar los entregables de cualquier sprint. Las habilidades necesarias para ser desarrollador es tener alto grado de dominio en el trabajo. Las responsabilidades que tienen suelen ser; Planificar y crear Sprint, adaptar los planes al Sprint, responsabilidad de uno a otros, añadir calidad (Schwaber & Sutherland, 2020).

Dueño del Producto (Product Owner)

El propietario del producto es el encargado de guiar el proyecto, conoce cuál es la dirección del proyecto, es el responsable de clasificar el trabajo según el índice de dificultad o del valor comercial. El propietario del producto representa las necesidades del cliente y las partes interesadas. Las decisiones que toma el propietario del producto deben ser respetados, la persona que tome el rol de Product Owner debe tener experiencia en la gestión de proyectos.

Es encargado de crear la lista de productos (Product Backlog), es responsable de la eficacia de Product Backlog los cuales incluyen (Schwaber & Sutherland, 2020);

- La lista de productos debe ser entendible y transparente sin generar confusión.
- Debe ordenar la lista de los productos.
- Crear y difundir los elementos de la lista de productos.
- Comunicar el objetivo del producto.

Líder del Producto (Scrum Master)

El Scrum Master es el encargado de aplicar Scrum al equipo tal como se define en la guía de SCRUM. Debe ayudar a entender la teoría y práctica del marco de trabajo Scrum. El Scrum Master es el responsable de que se aplique de manera efectiva el marco de trabajo Scrum.

El Scrum Master ayuda a los miembros del equipo de diferentes maneras (Schwaber & Sutherland, 2020):

- Entrenar a los miembros del equipo en funcionalidad cruzada y auto gestión.
- Ayudar a los miembros del equipo a cumplir con sus tareas.
- Eliminar los impedimentos para el progreso del Grupo de trabajo.
- Hacer cumplir los eventos de Scrum dentro de un límite de tiempo.

El Scrum Master ayuda al dueño del producto de varias maneras (Schwaber & Sutherland, 2020).

- Ayuda a encontrar técnicas para lograr un producto eficaz y gestión de lista de productos.
- Ayuda a entender las necesidades de manera clara y concisa las listas de productos.
- Ayuda establecer una planificación empírica de productos en los entornos complejos.
- Facilita la colaboración con los interesados según las necesidades.

1.2.5. Eventos en SCRUM

La finalidad de los eventos es minimizar y regularizar las reuniones que no están definidas en Scrum. Si no se realizan los eventos de Scrum produce las pérdidas de oportunidades para adaptar e inspeccionar.

El enfoque principal de los eventos es la transparencia, estos eventos tiene una duración máxima establecida. Existe cuatro eventos: Spring Planning, Sprint Review, Daily Scrum y Sprint Retrospective. Estos eventos se encuentran dentro de un Sprint.

Sprint

El sprint es núcleo de scrum, es como el corazón de los seres vivos. El sprint se lleva a cabo en un periodo de un mes o menos, se revisa las actividades y productos realizados del anterior sprint. Un nuevo Sprint inicia cuando se revisa y concluye el sprint anterior (Gonçalves, 2018). Durante la ejecución del Sprint no se deben de realizar modificaciones, esto puede afectar al objetivo principal de un Sprint. Un Sprint puede ser cancelado si el objetivo del Sprint es obsoleto, solo el dueño del producto tiene la autorización de cancelar el Sprint.

El autor (Schwaber & Sutherland, 2020) menciona que durante el Sprint pueden suceder los siguientes casos:

- No realizar cambios esto puede provocar cambios en el objetivo del Sprint
- La calidad no debe disminuir.
- La lista de productos se refina según sea necesario.

- El alcance se clarifica y puede ser renegociable a menudo que se va aprendiendo más.

Sprint Planning

El Sprint Planning reúne a todos los miembros del equipo. El equipo define lo que se puede desarrollar durante el Sprint, también pueden invitar a otras personas para brindar asesoramiento. El propietario del producto revisa el objetivo que desean cumplir con el Sprint, así como también la lista de productos que conducen al objetivo. El equipo de Desarrollo ejecuta el Sprint, siempre teniendo en cuenta la meta del Sprint (Gonçalves, 2018).

El Sprint Planning aborda varios tópicos que respondes a las siguientes preguntas (Schwaber & Sutherland, 2020): ¿Por qué el Sprint es valioso?, ¿Qué se puede hacer durante el Sprint? y ¿Cómo se realizara el trabajo planteado en el Sprint?

La planificación de un Sprint se lleva a cabo en un límite de tiempo de no más de ocho horas para un Sprint de un mes. Cuando un Sprint es corto, los eventos son más cortos depende de cuánto tiempo tomo realizar el Sprint.

Daily Scrum

Daily Scrum tiene como propósito inspeccionar el progreso hacia el objetivo del Sprint, adaptar el Product Backlog según sea necesario ajustando para el próximo trabajo. La duración de Daily Scrum es de 15 minutos para desarrolladores del equipo de trabajo. Para disminuir las dificultades, las reuniones se llevan a cabo en la misma hora y el mismo lugar. Si el Product Owner o el Scrum Master están trabajando activamente en los ítems del Sprint Backlog, estos participan como desarrolladores.

En Daily Scrum involucra la revisión de trabajos que se ha realizado des la última Daily Scrum. Ayuda a mejorar la comunicación identificar impedimentos, promueve la toma de decisiones rápido y elimina las necesidades de las reuniones anteriores. El Daily Scrum no es el único momento en que los desarrolladores pueden reunirse, también pueden reunirse a lo largo del día para posibles inquietudes o pedir más detalles del Sprint (Schwaber & Sutherland, 2020).

Sprint Review

Una vez que se hay concluido con el Sprint, es necesario realizar un Sprint Review. Es atendido por el Scrum Team y Stakeholders quien es invitado por el Product Owner. El Sprint Review tiene una duración aproximada de 4 horas en el caso de que el Sprint duro un mes. Se realiza una revisión para detectar que es lo que se ha logrado durante el Sprint, revisión de las tareas de Sprint Review anterior.

Después de Sprint Review el Product Backlog puede ser reajustado para satisfacer nuevas necesidades. Se revisan los elementos probables del Product Backlog para el próximo Sprint (Schwaber & Sutherland, 2020).

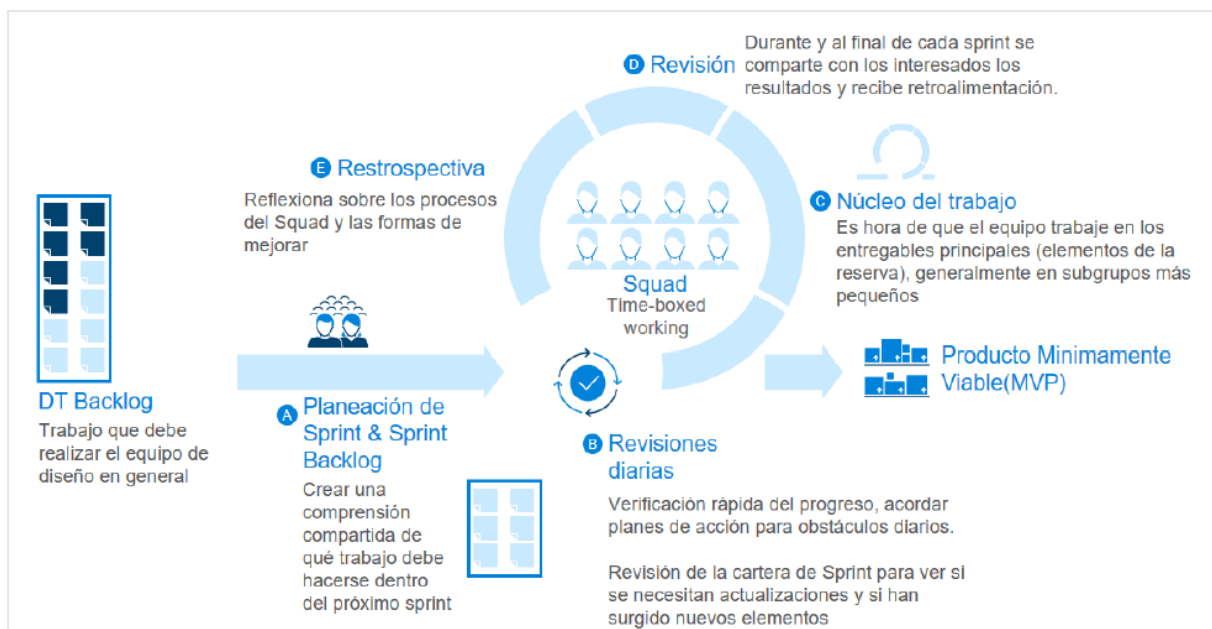
Spring Retrospective

El objetivo de Spring Retrospective es planear un incremento de calidad y eficacia. El Team Scrum investiga cuáles fueron los resultados del último Sprint con respecto a las personas, los procesos, las iteraciones, las herramientas. Estas inspecciones pueden variar dependiendo del dominio del trabajo. Se exploran cuáles fueron los inconvenientes que llevaron a un mal camino, que dificultades tuvieron, identifica los cambios más útiles que mejoraron su efectividad. Las mejoras que tuvieron buenos resultados se pueden incorporar al Sprint Backlog.

El Sprint Retrospective tiene una duración de 3 horas en el caso de que el Sprint haya durado un mes, en otros Sprints cortos requiere una reunión corta. Este Sprint Retrospective es esencial para centrarse en inspección, adaptación y mejoramiento (Gonçalves, 2018).

Figura 8

Fases de spring.



Nota. Fuente (Hadida & Troilo, 2020).

1.2.6 Artefactos de Scrum

Los artefactos de Scrum son representación del esfuerzo empleado en el trabajo, el valor del trabajo. El objetivo principal de los artefactos de Scrum es proveer transparencia a través de

la comprensión compartida de trabajo. Brinda a los equipos con oportunidades de adaptación e inspección (Gonçalves, 2018).

Los Artefactos se rigen mediante compromisos para asegurar que proporcionen información que mejoren la transparencia y el enfoque con esto se puede medir los procesos (Schwaber & Sutherland, 2020).

- Para el Product Backlog está el Product Goal
- Para el Sprint Backlog está el Sprint Goal
- Para el incremento está la definición de hecho (Definition of Done)

Product Backlog

Es una lista de todos los requerimientos que deben cumplir el producto final, también incluyen las modificaciones que se realicen durante el desarrollo del producto. La lista de producto tiene una descripción, estimación, orden y valor (Gonçalves, 2018).

Los elementos de un Product Backlog son realizados por el equipo de trabajo, listo para que se tomen en la planificación de un evento de Sprint. Los desarrolladores involucrados en la creación del Product Backlog son responsables del dimensionamiento, también pueden estar involucrado el Product Owner ayudando a comprender y seleccionar compensaciones.

Compromiso Product Goal: describe un estado futuro del producto, esto sirve como objetivo para que el equipo Scrum pueda planificar. Todo el Product Backlog debe emerger para cumplir con el Product Goal (Schwaber & Sutherland, 2020).

Sprint Backlog

El Sprint Backlog es una lista de los elementos de trabajo, usualmente se le suele llamar User Stories, que se suelen realizar dentro de un Sprint. Si se desea realizar seguimiento de trabajo, se pueden hacer uso de gráficos de los trabajos pendientes (Gonçalves, 2018). Generalmente de un Sprint anterior se puede realizar mejores al Sprint Backlog según como se vaya aprendiendo, esto puede ayudar en la inspección del progreso en Daily Scrum.

Compromiso Sprint Goal: Brinda flexibilidad, también crea enfoque y coherencia animando al equipo a trabajar de manera grupal en lugar de iniciativas independientes. El

Sprint Goal se crea en el Sprint Planning y posteriormente se agrega al Sprint Backlog (Schwaber & Sutherland, 2020).

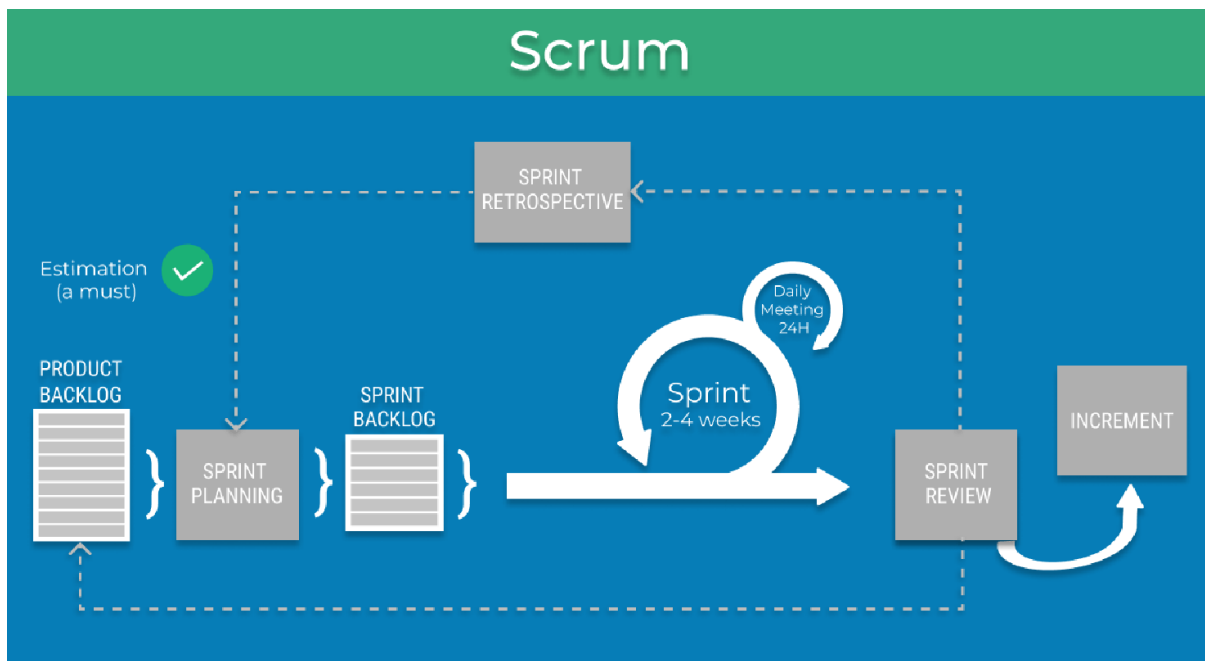
Increment

Se define como incremento cuando al finalizar el Sprint cumple con la necesidad del cliente, el incremento se añade al Sprint Backlog. Se pueden crear múltiples incrementos dentro de un Sprint. El acumulativo de los incrementos se presenta en el Sprint Review. Se pueden entregar los incrementos a la parte interesada antes de que se termine el Sprint.

Compromiso Definition of Done: Se considera terminado cuando cumple con las medidas de calidad requeridas para el producto. En el momento en el que un elemento del Product Backlog se considere listo, se crea un incremento. En caso de que un elemento no se haya cumplido en un Sprint, vuelve a agregarse al Product Backlog para realizarse en otros Sprints. La definición de terminado es proporcionada por el grupo (Schwaber & Sutherland, 2020).

Figura 9

Fases de scrum



Nota. Fuente (Schwaber & Sutherland, 2020).

1.3. Herramienta GitHub

1.3.1. Origen de GitHub “Git”

Git es un controlador de versión de sistemas. Que permite obtener la trazabilidad de un desarrollo de software y diseño de software. Git es una distribución de control de versiones de sistemas, todo un equipo de trabajo trabaja sobre un mismo proyecto. Se trata de una del historial de cambios realizados sobre el trabajo o proyecto, no solo el estado actual del archivo (Brent, 2018).

Control de versión de sistemas: Es un sistema capaz de registra los cambios realizados en archivo o varios archivos que pertenecen a un proyecto. Se puede volver en el tiempo desde el futuro para recuperar una versión específica de un archivo. Cuando se instala un controlador de versiones de sistemas se instala de manera local en el mismo lugar que se encuentre el archivo o proyecto (Ravishankar, 2013).

Para entender mejor, el control de versión de sistemas es un paquete donde se registra las actividades que se van realizando, el registro de actividades se almacena en diferentes niveles con una etiqueta para que pueda volver a visitar o editar esos niveles etiquetados cuando se necesite.

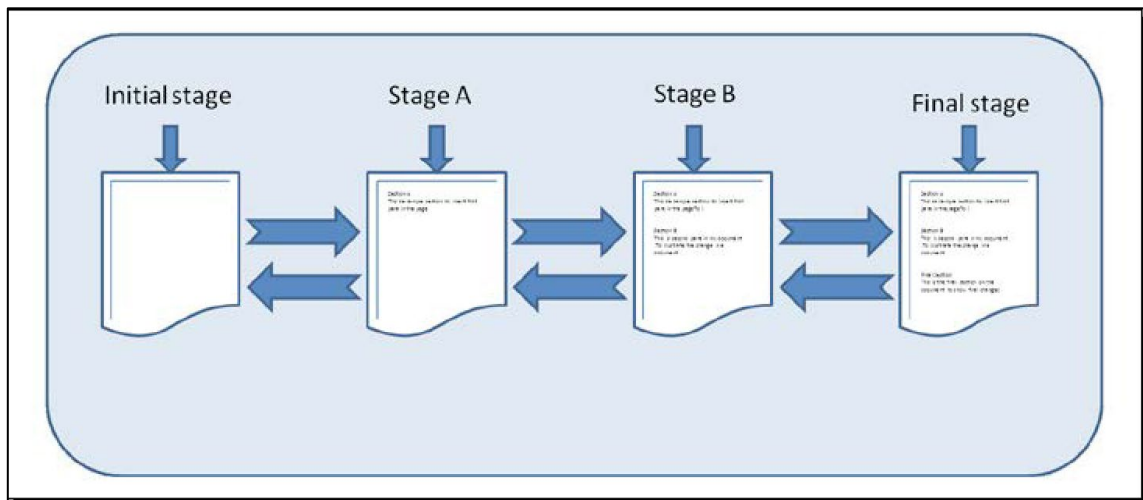
El autor (Ravishankar, 2013) menciona que las características del control de versiones de sistemas son:

- Se puede tener varias versiones del mismo archivo con el mismo nombre, con esto se evita el desorden con los nombres de los archivos.
- Tiene la facilidad de marcar una parte específica para que pueda usar en el futuro antes de las modificaciones.
- Es fácil de recuperar versiones anteriores, modificado por diferentes autores.

En el control de versiones de sistema se registra el punto de partida, esto es de ayuda para cuando se desea volver a la versión anterior. Es decir, se registra los números de etapas, se considera etapas al punto donde comienza la edición hasta que se guarda y finaliza.

Figura 10

Almacenamiento de etapas en GitHub.



Nota. Fuente (Ravishankar, 2013).

Sistemas De Control De Versiones Distribuidos (SCVD)

Este sistema es el mejor de los otros dos sistemas de control de versión local y sistema de control de versión centralizado. El sistema de control de tipo distribuido combina las mejores partes de los dos, por un lado, tenemos el almacenamiento de manera local guarda las etapas de las versiones en cada máquina del usuario si así fuera el caso. Por otro lado, almacena las versiones en un servidor de manera remota.

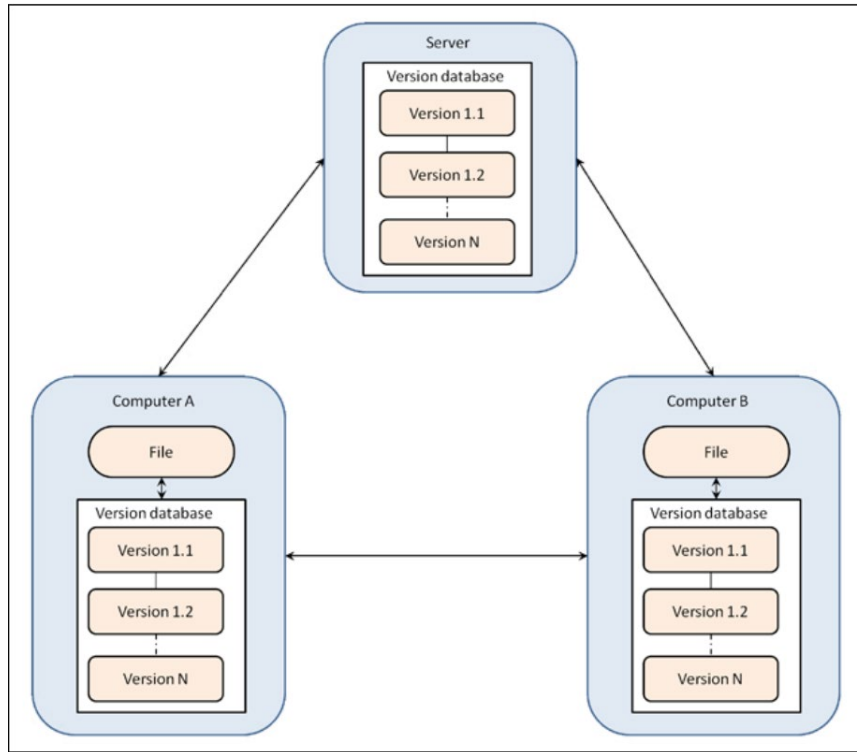
En este tipo **SVCD** el historial de las versiones se encuentra a salvo, si el servidor se corrompe se tiene respaldos de manera local en las máquinas de los que estén trabajando en el archivo. Sí por el contrario se corrompe las maquinas locales de los usuarios, se tiene un respaldo en el servidor.

Con el sistema de control de versiones distribuidos tenemos varias ventajas, algunas de las ventajas se combinan con las ventajas del sistema de control de versiones centralizados. Las ventajas son:

- Realización de cambios locales sin preocuparse por la conectividad de tiempo completo con el servidor.
- No depender de una sola copia de los archivos almacenados en el servidor.
- Reutilización del trabajo
- Trabajo colaborativo, sin depender del historial almacenado en máquinas individuales.

Figura 11

Sistemas de control de versiones distribuidas.



Nota Fuente (Ravishankar, 2013).

1.3.2. ¿Qué es GitHub?

GitHub ofrece varias funcionalidades que facilitan la comunicación y colaboración dentro de un equipo. Su característica más importante es el mecanismo de solicitud de extracción, que es una forma de iniciar una discusión con otros usuarios y compartir o comentar sobre los diversos artefactos en un proyecto, en las discusiones se tratan de cambio en los contenidos del proyecto. La discusión puede convertirse en una tarea o actividad a realizar (Zagalsky et al., 2015).

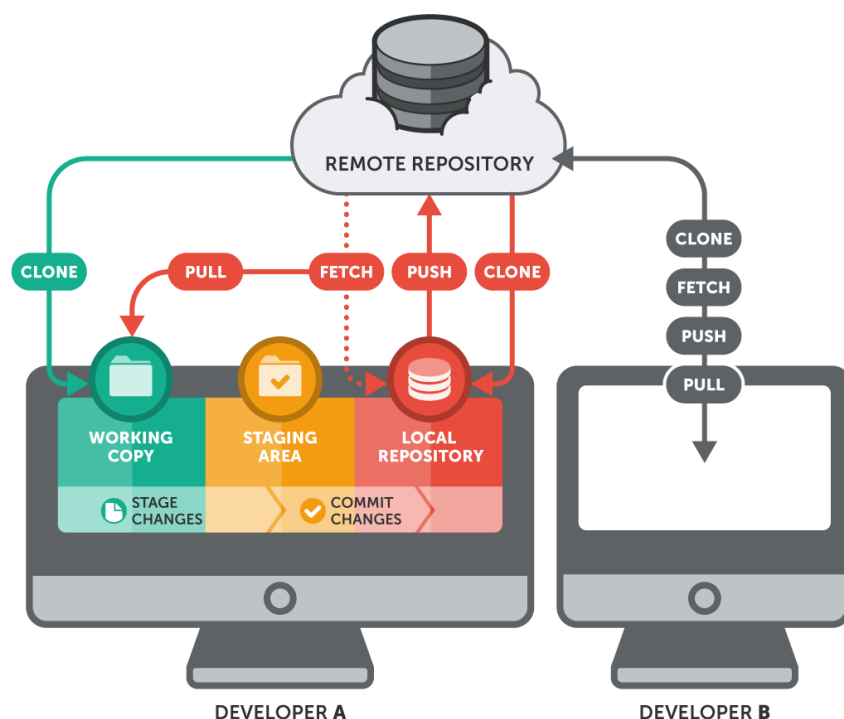
La discusión puede incluir un código que es visible para todo y mostrarse exactamente los cambios que se realizaron si se acepta la discusión. También puede incluir otro contenido (p. ej., capturas de pantalla) para proporcionar antecedentes para la discusión o incluir cambios en otros recursos del proyecto.

GitHub también apoya la concientización mediante la transmisión de actualizaciones a las noticias del usuario. La combinación de estas características facilita “una cultura de colaboración espontánea pero estructurada.

GitHub es beneficioso para la colaboración de una comunidad debido a la transparencia, donde los contribuyentes individuales pueden inferir los objetivos técnicos de un colaborador o estar al tanto de lo que otros usuarios están prestando atención.

Si un usuario desea contribuir en alguna parte del proyecto, él interesado puede clonar el proyecto hacia su repositorio, esto crea una copia completa del proyecto al entorno local. Las modificaciones o cambios comprometidos seguirán realizando cambios en el repositorio original. Sí el colaborador no quiere hacer cambio en el repositorio original puede realizar una bifurcación a esto se lo llama Fork. Otra alternativa para no realizar cambios al archivo original es la creación de ramas para el colaborador, todos los cambios se almacenan en la rama.

Figura 12
Arquitectura GitHub.



Fuente: (Velmurugan, 2021)

1.3.3. Estado de GitHub en la actualidad

En la actualidad GitHub es una de las herramientas más usados en trabajos colaborativos sobre todo en proyectos enfocados al desarrollo del software. Cuenta con grandes ventajas

para la gestión del software, entre sus principales ventajas es el control de versiones. Se ha establecido como la plataforma más importante para trabajo colaborativo, es un software libre (Zöller et al., 2020).

La plataforma de GitHub presta servicios de productos libres y de pago, permite guardar y colaborar en el código de un proyecto. Los servicios pueden ser personal, organizacional y empresarias. El número de colaboradores en el repositorio es ilimitado.

Los repositorios de GitHub se utilizan para realizar trazabilidad y almacenar información sobre artículos académicos centrados en las carreras de ingeniería mayoritariamente aquellos centrados en la construcción de software. Se realizan referencias a repositorios de acceso público, la mayoría de estos enlaces son influyentes para la actividad académica, promueven e incentiva a la investigación e información compartida. Los enlaces provienen de **Readme.md**, es un recurso para la documentación, es una carta de presentación de los proyectos alojados en GitHub (Wattanakriengkrai et al., 2022).

GitHub es la herramienta que está ayudando al movimiento de código abierto, de hecho, en la actualidad las personas confían en las bibliotecas y marco de código abierto. GitHub está contribuyendo al desarrollo de código abierto en un mercado competitivo en el que el 86% de los desarrolladores hacen uso del código abierto (Coelho et al., 2020).

1.3.4. ISO/ IEC 25022

Medición calidad en uso, proporciona un conjunto y estructura para evaluar la calidad del producto, definen conjunto de métricas y funciones que son útiles para evaluar cada característica del software, estas características son: Funcionalidad, usabilidad, eficiencia, efectividad y mantenibilidad.

La ISO 25022 se enfoca de llevar cabo evaluaciones mediante métricas, estas métricas pueden ser modificadas, siempre que no se respete el estándar de la calidad en uso y calidad interna, externa. Este estándar es parte de la ISO 25000 el cual tiene como propósito la evaluación de software.

En la Tabla, se presenta el formato de las especificaciones de la ISO/IEC 25022 de la métrica de usabilidad del producto, referenciando los valores para cada variable.

Tabla 8

Medidas de efectividad.

Descripción	Formula	Valor deseado	Tipo de medida
Proporción de tareas que se completan correctamente	$X=A/B$	$0 \leq X \leq 1$ Cuanto más cercano a 1, mejor.	A=Contable, número de tareas únicas completadas B=Contable, número total de tareas únicas intentadas.

Nota. Fuente (ISO/IEC 25022, 2016).

CAPÍTULO II

2. Diseño de la propuesta

Este capítulo se estable un marco de trabajo que fue de ayuda para realizar el diseño de la guía metodológica haciendo uso de GitHub y SCRUM. Esta sección está dividida en: Metodología para desarrollar la guía metodológica y diseño de la guía metodológica.

2.1. Metodología para desarrollar la guía Metodológica

2.1.1. Definición de la Guía Metodológica.

Una guía metodológica es una plataforma que permite la comunicación y la documentación del proyecto. Aumenta la interacción entre los miembros del equipo permitiendo el entendimiento del objetivo que se desea cumplir al desarrollar un producto o servicio. Otro de los beneficios que se logra al tener una guía es que obtiene un estándar para que no se olvide de las tareas que deben cumplirse.

Cuando una persona tiene una guía nunca deja de documentar las actividades que va cumpliendo, siempre está atento a las actividades que debe cumplir antes de finalizar con la tarea. Una buena Guía garantiza que las actividades se realicen correctamente y no se pierda pasos vitales que afecten a la calidad del software. Permite construir los datos y las actividades de un proyecto de forma estructurada, esto facilita la revisión de actividades a futuro.

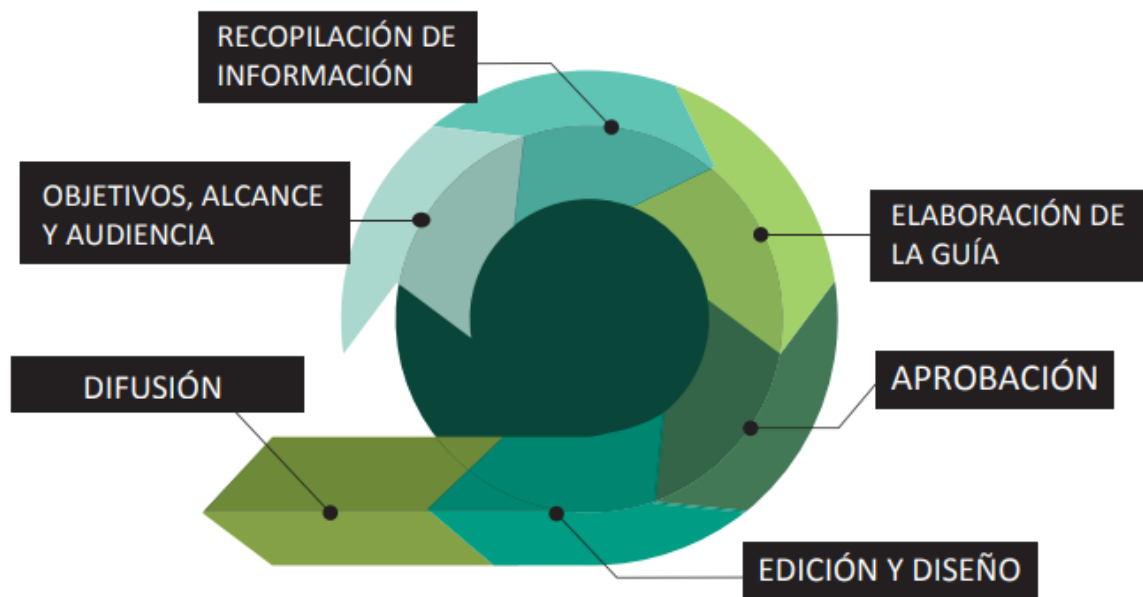
Se define como guía metodológica a un documento en el que se describe un conjunto de normas a seguir. Las Guías Metodológicas están relacionados con proyectos o sistemas de información. Las Guías enlistan los pasos que se deben seguir en orden lógico. Las Guías deben de ser probadas para sustentar la efectividad y éxito que se obtienen al aplicar.

2.1.2. Fases de la Guía Metodológica

La guía está estructurada por una serie de fases, comenzando con la definición del alcance, objetivos y determinar la audiencia a quien se dirige la guía. A continuación, es necesario recopilar información relacionado con el objetivo planteado. Una vez recopilada la información está listo para elaborar la guía. La guía necesita ser validada y aprobada, para después pasar por un diseño y edición. Finalmente se procede a difundir la guía. La Figura 13, muestra las fases de la Guía Metodológica.

Figura 13

Fases para desarrollar una guía metodológica



Nota. Fuente (Robles Belmonte, 2017).

Paso 1. Objetivo, Alcance y Audiencia

Esta fase se enfoca en el objetivo, alcance y audiencia que el tema debe abarcar. Para lo cual es necesario plantearse las siguientes preguntas:

¿Qué se desea conseguir u obtener con la guía? Con esta pregunta se busca encontrar el objetivo principal o lo que se desea conseguir con la elaboración de la guía. Para esto es necesario, buscar el problema a solucionar. Se debe realizar la documentación pertinente a esta fase. Tras responder estas preguntas se obtiene como resultado el **Objetivo**.

¿Cuál es la audiencia objetivo? ¿Para quién es la Guía Metodológica? ¿Quién está interesado y por qué? Es de suma importancia identificar el público objetivo al que se desea llegar con la guía. Debe existir partes interesadas en la guía, se debe solucionar un problema específico. Con estas preguntas se encuentra la **Audiencia** a quien se dirige la guía.

¿Qué conocimiento se quiere normalizar? Es importante tener en claro cuáles son los contenidos que se van a incluir en la guía. En este caso se van a incluir el uso de las herramientas de GitHub y el marco de trabajo Scrum. Con esta pregunta se obtiene el **Alcance**.

Paso 2. Recopilación de la información.

Esta fase se centra en realizar una búsqueda de las herramientas necesarias para poder realizar la guía. Para poder seleccionar la herramienta correcta se debe hacer un análisis exhaustivo. Entre las herramientas a seleccionar pueden estar:

- Documentos de proyecto. Informes de misión y visión de la empresa, requerimientos de los usuarios, informes de algunos proyectos, históricos de donantes, vistas realizadas a las empresas y procesos de las empresas.
- Reuniones o grupos de trabajo, discusiones con jefes, especialista, interesados e involucrados en la materia.
- Encuestas, cuestionarios o entrevistas realizados. Son de utilidad para recopilación de la información.
- Otros documentos que se han semejantes a la guía que se desea realizar.
- Documentos de fuentes externas que aporten conocimiento al proyecto.

Este documento debe ser documentado y registrado, producir un documento de prototipo o borrador que se tendrá a la mano para realizar el documento final. Es documento no debe exceder de 2 páginas (Robles Belmonte, 2017).

Paso 3. Elaboración de la Guía.

A continuación, se presenta los apartados con el que debe contar la guía metodológica mencionado por (Robles Belmonte, 2017):

Prologo: En esta parte se detallan las motivaciones por el cual se realizó la guía, también se detallan los aspectos más importantes que se consideren para la interpretación de la guía. Expone las principales aportaciones.

Resumen ejecutivo: En este apartado se debe especificar cual es la importancia de la guía, que finalidad se tiene, cual es el objetivo para cumplir. Se expone de manera clara cuál es el alcance de la guía.

Introducción: En este apartado se debe presentar el tema a tratar. Se trata de presentar la guía y no resumir se debe presentar los siguientes puntos: Antecedentes, ubicar la guía en espacio y tiempo, abordar el problema de la investigación y exponer el propósito general (Robles Belmonte, 2017).

La introducción debe contener: En que contexto se desarrolla la guía, los antecedentes y las problemáticas que soluciona el proyecto y los objetivos de la guía. Para los objetivos es recomendable tener los resultados que se han obtenido en cuanto a sus beneficios.

Cuerpo de la Guía: Esta es la parte más importante, debe especificar cuáles son los pasos que se deben realizar durante el proceso o metodología. Dentro de esta sección se deben de tener: El enfoque metodológico, identificación de la fases principales e identificación de las actividades que se realizan en cada una de las fases, esto se debe realizar de manera sistemática.

Las actividades se deben presentar de tal manera que se entienda, no exista confusión al seguir la guía. En las actividades se deben mostrar:

- Que se desea lograr con la actividad
- Detallar las actividades que deben realizar.
- Cuáles son las técnicas y las herramientas que deben utilizar.
- Los resultados que se desean obtener.
- Personas involucradas.

Conclusiones y consideraciones: En esta parte se destacan cuáles son los puntos clave. Se deben detallar cuales son las lecciones que se aprendió, cuales fueron los resultados. También se pueden mencionar las recomendaciones para implementar la guía. Para la documentación se debe tener en cuenta:

- Las lecciones deben dar a conocer los resultados y cuáles fueron los factores que infligieron el éxito o la obstaculización del desarrollo de la guía.
- Se deben identificar la causa y efecto según el contexto.
- Las lecciones deben de proporcionar recomendaciones.

Revisión: Luego de que se haya desarrollado la guía se debe realizar una revisión. Para realizar la revisión se recomienda seguir los siguientes pasos:

- Revisar la documentación en cuanto a la información incluida en la guía, si existe citas es recomendable revisar si están citados correctamente. No es necesario incluir todo el texto de las citas, si no únicamente las partes que contribuyeron a la realización de la guía.

- Es recomendable que se incluya a personas externas para la revisión de la documentación. Las personas externas pueden debatir acerca del proyecto o revisión de sintaxis.
- Es necesario contar con experto en el área, para que comente de un modo crítico y valide los resultados.

Paso 4. Aprobación.

Después de desarrollar la guía es necesario que sea aprobado por una persona u organización, la persona responsable de la aprobación debe ser un especialista con conocimientos relacionados a la guía. No se puede concluir con esta fase sin antes revisar detenidamente la documentación de la guía.

Paso 5. Edición y diseño.

Posterior a la revisión es necesario que se envíe el documento a una revisión de gramática y ortográfica. Verificar las tipografías correspondientes, se debe diseñar el documento tomando como ejemplo una plantilla recomendada por la persona validadora.

Paso 6. Difusión y Medición.

Tras terminar la guía y que haya sido aprobado llega a difusión. Es el último paso para realizar, es necesario buscar los canales de difusión que permitan llegar a la audiencia objetiva. Los canales de difusión pueden ser revistas, repositorios bibliográficos, talleres y ponencias.

En cuanto a la medición de la guía se utilizaron la ISO/IEC 25022 enfocado en la medición de las funciones que se plantearon como objetivo de cada fase. Dicha función permitió saber la calidad definida en las guías y la complejidad de aplicar la guía.

2.1.3. Diseño de la Guía metodológica

La guía metodológica de software se enfocó en desarrollar una guía para la gestión de proyectos que permita tener una trazabilidad en la creación de un software. Por lo cual se tomó como eje principal la aplicación del marco de trabajo Scrum. Para permitir la trazabilidad y gestión de un proyecto de software se realizó en tres fases Pre-Juego, Juego y Post-Juego, dichas fases fueron obtenidas de Scrum.

Para cumplir con los objetivos de las fases de scrum se hizo uso de la herramienta GitHub, es una herramienta que permite el control y gestión de proyectos de software. La herramienta utiliza el control de versiones de código el cual encaja perfectamente con las fases de Scrum que son iterativas.

La guía metodológica que se desarrollo tiene como nombre **Scrum-GitHub** ya que es la unión del marco de trabajo Scrum y la herramienta GitHub.

2.1.4. Artefactos y Eventos de Scrum, correspondencia con las funcionalidades de GitHub

Cada uno de los artefactos y eventos de scrum se gestionan con las funcionalidades de GitHub. La Tabla 9 presenta las respectivas correspondencias de eventos y artefactos frente a las funcionalidades de GitHub.

Tabla 9

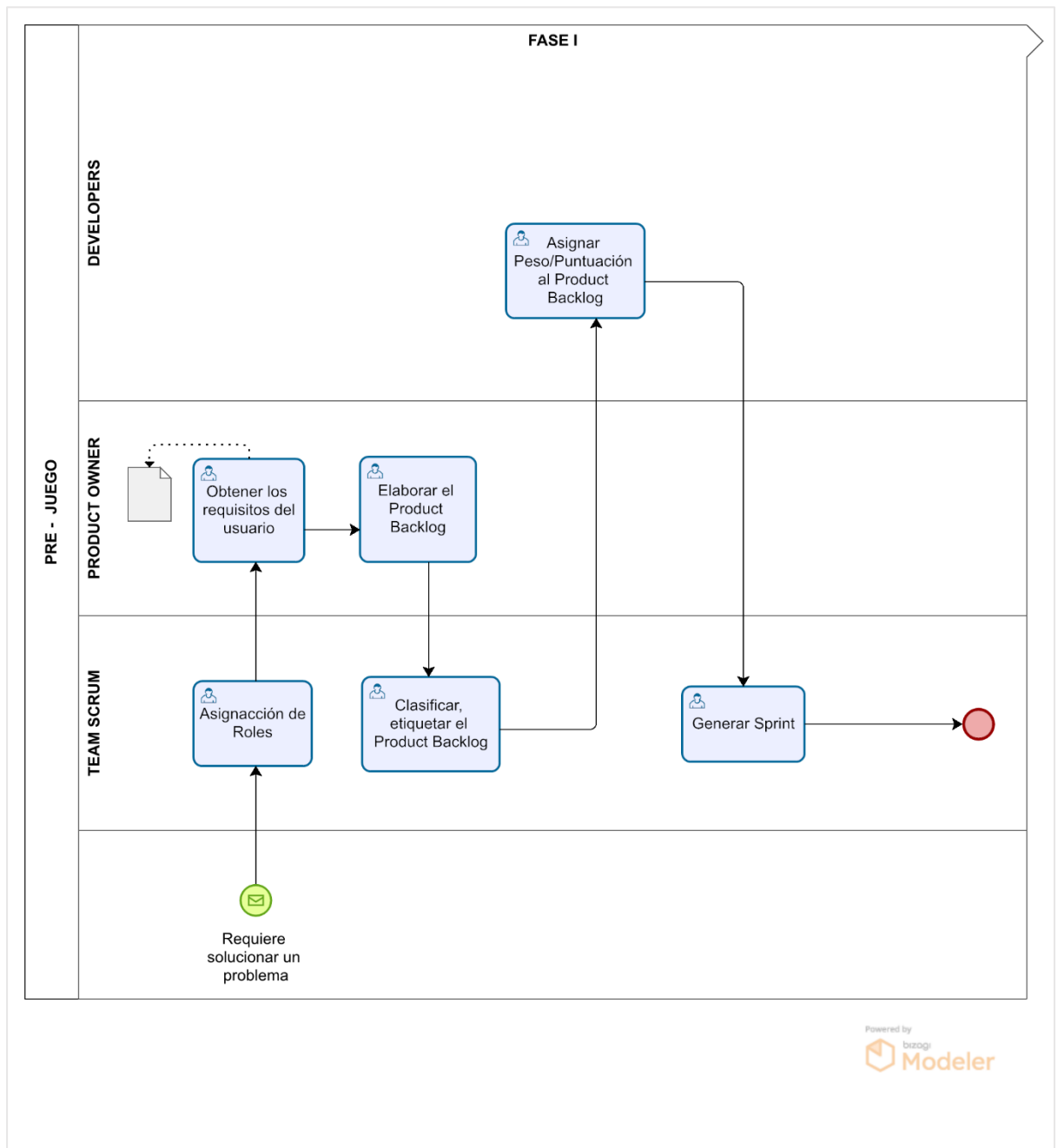
Artefactos y eventos de Scrum con las funcionalidades de GitHub.

Fase	Actividad Scrum	Tarea Scrum	Funcionalidades de GitHub
	Inicio	Creación de Proyecto, planeación	Repository
Pre-Juego	Sprint 0	Roles	Collaborators
	Levantamiento Requerimiento	Product Backlog	Issues
Juego	Sprint	Sprint Backlog	Iteration Project, Milestones
		Daily Scrum	Discussions, Issues
		Sprint Review	Discussions, Issues
		Retrospectativa	Discussions, Issues
Post-Juego	Incremento	Increment	Tags
	Publicación	Publisher	Pages, Releases
	Mantenimiento		Discussions, Issues

2.1.5. Pre-Juego

La primera fase es el inicio del proyecto también llamado el Sprint 0. En esta fase se informa de la necesidad o problema a resolver. Es el punto de partida en el que se conforma el grupo de trabajo y se asigna los roles a cada miembro e interesados en el proyecto.

Figura 14
Fase 1 Pre-Juego



Esta fase es el inicio del proyecto, empieza con la detección del problema a solucionar. Ante una problemática se procede a conformar un grupo de personas capaces de dar solución a la necesidad encontrada. Par esto los miembros del equipo deben empezar asignando roles entre ellos, los roles que se asignan se definen en el maco de trabajo Scrum detallado en la Figura 7.

Una vez que se haya asignado los roles, el Product Owner o también conocido como el dueño del producto es el encargado de obtener los requisitos del usuario, debe formar un historial

de usuarios, la unión de varias listas de usuarios se conoce como Product Backlog dentro de Scrum. Product Backlog es gestionado por producto Owner.

Cada requerimiento del Product Backlog debe ser informado a todos los miembros del equipo. Entre todos los miembros del equipo deben etiquetar o clasificar los requerimientos. Posteriormente los Developers son los encargados de asignar el peso de dificultad que tiene para desarrollar el producto o requerimiento.

Luego de que cada Product Backlog tenga el peso correspondiente se procede a elaborar el Sprint. Un Sprint está conformado por varios requisitos que tiene un objetivo en común, el Sprint tiene un objetivo que cumplir. Los objetivos de Sprint deben ser informados a todos los miembros del equipo.

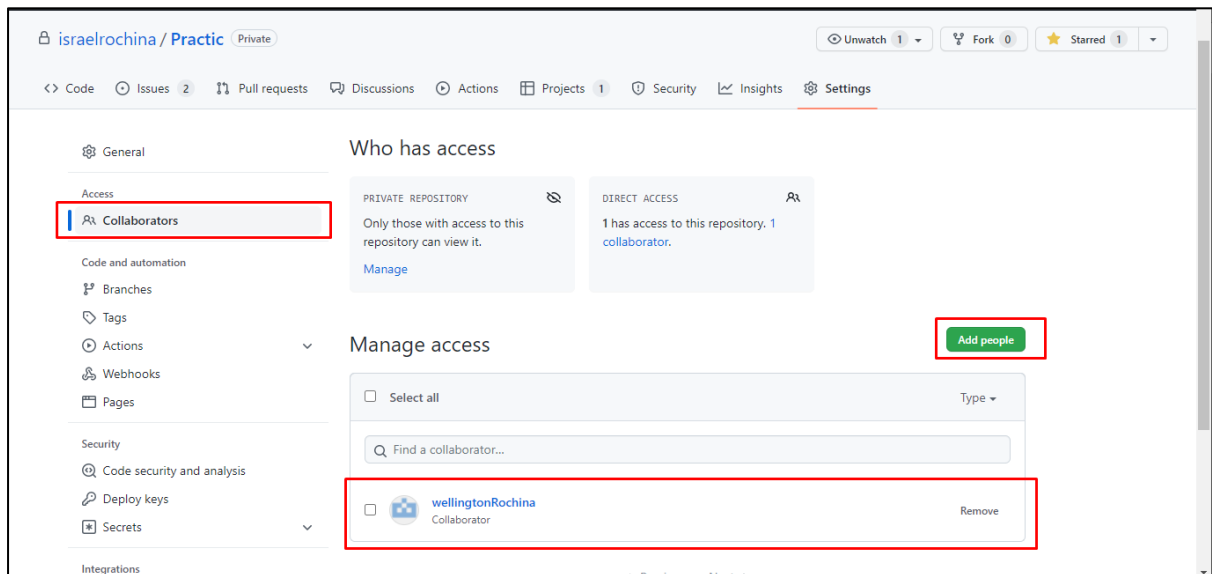
Collaborators - Roles

Dentro de la herramienta de GitHub es posible unir a varios miembros que deseen colaborar en un proyecto. La funcionalidad de unir colaborador (Collaborators) permite agregar colaboradores al repositorio de trabajo mediante el nombre de usuario GitHub como se observa en la Figura 15, los colaboradores tienen dos tipos de rol como administrador y miembro.

Administrador: Al estar como administrador tiene el acceso completo a realizar configuraciones en el repositorio y también puede incluir más colaboradores.

Miembro: Puede ver todas las actividades del repositorio, pero no puede hacer configuraciones que alteren el repositorio.

Figura 15
Collaborators (Rol)



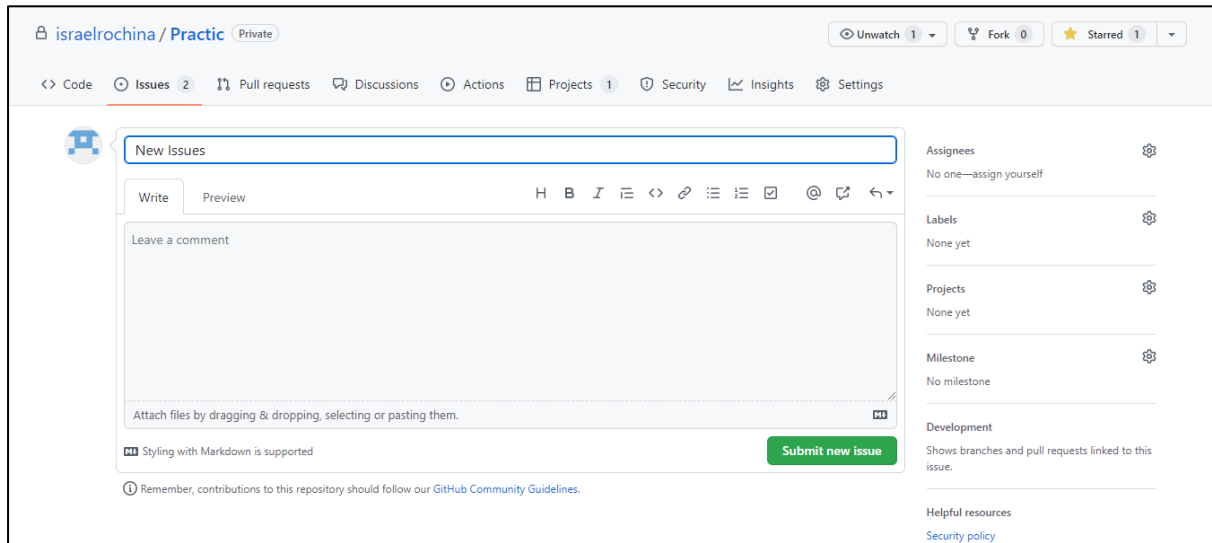
Issues – Product Backlog

En la herramienta de GitHub las Issues son actividades que parten de una discusión. Las discusiones se generan por diferentes motivos: Dudas sobre el proyecto, sugerencia de cambio de diseño, funcionalidades que va a tener el programa. Dentro de una discusión se forma un hilo de conversación que al final se pueden convertir en una Issue.

Con las Issues se puede crear el Product Backlog y así registrar todos los requerimientos del cliente. Además, las Issues tiene configuraciones personalizables que ayudan a gestionar los requerimientos como se observa en la Figura 16.

Figura 16

Issues



Labels (Etiquetas) – Clasificar por dificultad

Los labels es una manera de etiquetar las Issues que tiene GitHub, como se observa en la

Figura 17 las etiquetas son personalizables se pueden asignar el nombre que se desee. Los labels pueden ser usados de diferentes maneras para asignar un nivel de importancia a las Issue o para marcar a que grupo pertenece una Issue.

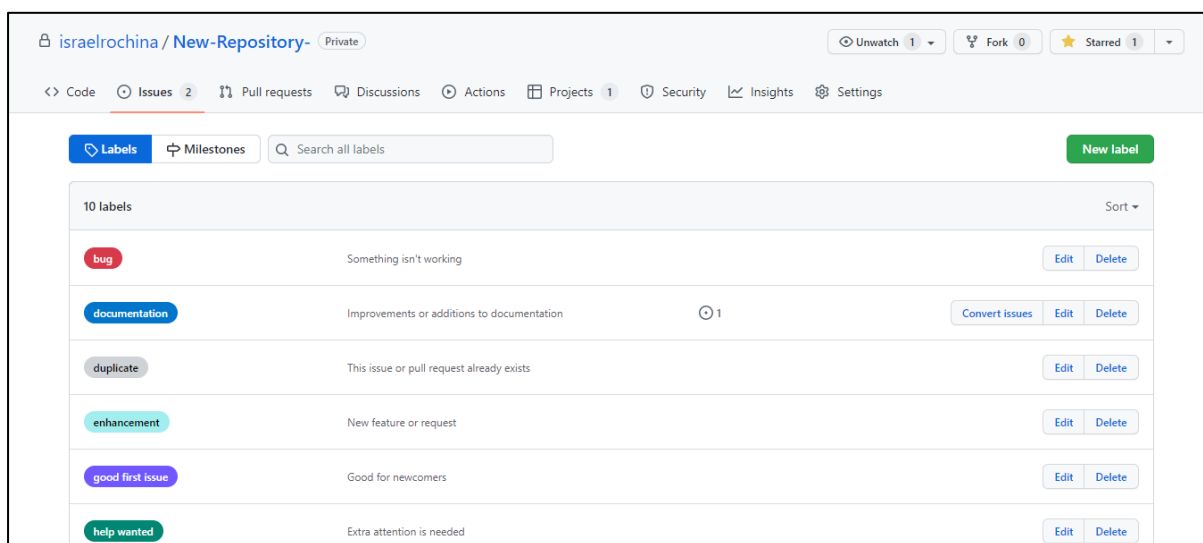
Con los labels se pueden asignar el grado de dificultad a cada uno de los Product Backlog de marco de trabajo Scrum.

Los labels facilitan:

- Hacer búsquedas por filtro a las Issues
- Realizar seguimientos a las Issues

Figura 17

Labels

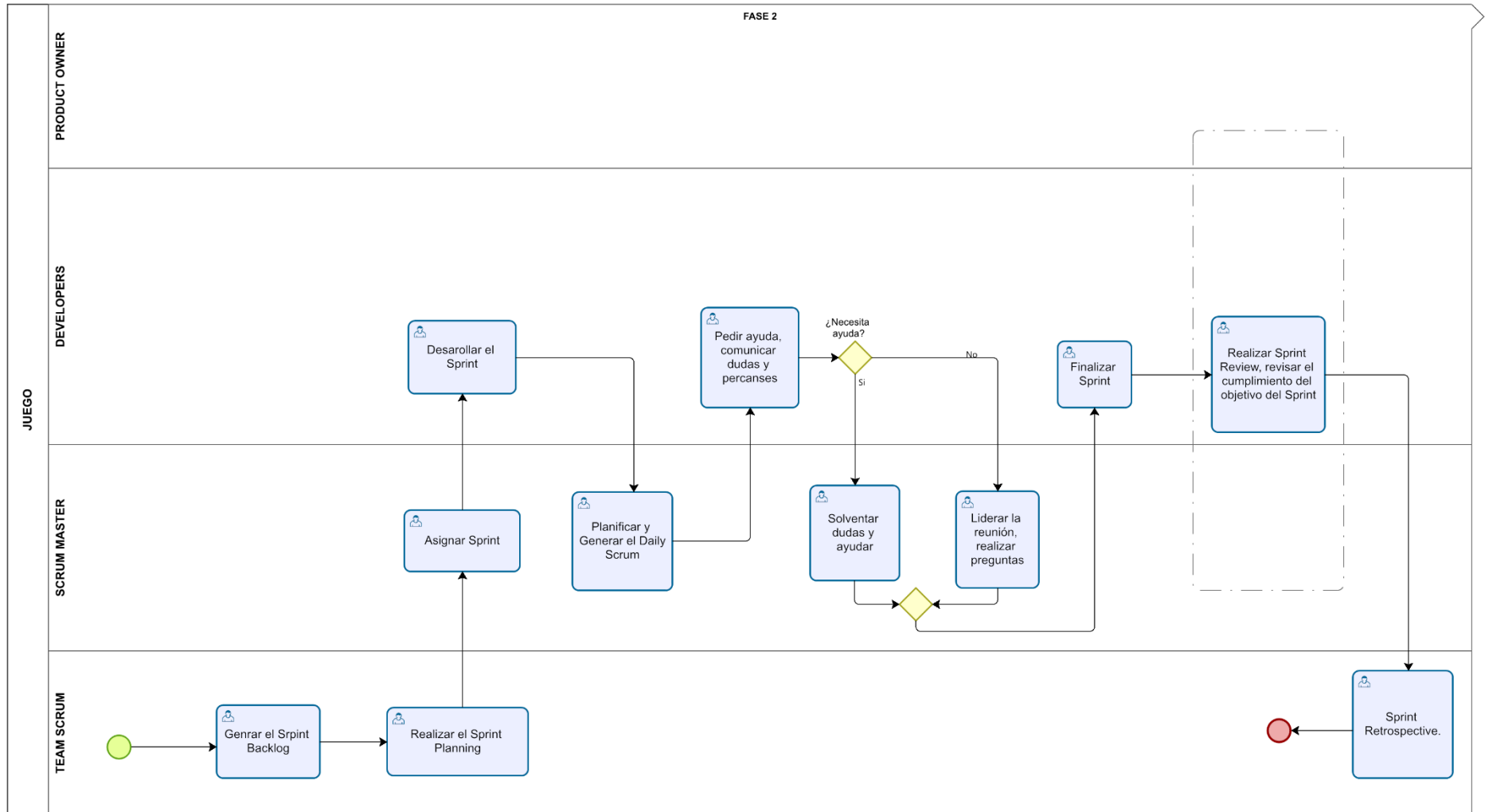


2.1.6. Juego

El juego es la segunda fase del marco de trabajo scrum, en esta fase se organiza los productos backlog. Se procede a crea los Sprint, cada Sprint tiene un objetivo y es asignado a un desarrollador como tarea personal.

Figura 18

Fase 2 Juego



Esta fase parte desde los product backlogs, una vez que se definan cuáles son los objetivos del product backlog se procede a crear el Sprint Backlog. Para lo cual el Team Scrum debe asignar el tiempo de desarrollo que tiene cada Sprint, el tiempo depende del nivel de dificultad de cada Sprint. Los tiempos de desarrollos de cada Sprint debe ser homogéneas entre todo los Sprint, esto ayuda a tener una mejor planificación para las próximas actividades y revisiones de avances.

Luego de haber creado el Sprint Backlog, el Scrum Master debe designar las tareas a los Developers. Los Sprint se asignan dependiendo del nivel de experiencia en desarrollar software. Con la primera asignación del Sprint se podrá estimar el tiempo que se demora una persona en completar y cumplir con el objetivo.

Una vez que cada Developer tenga asignado una tarea, procede a realizar hasta el tiempo que se asignó al Sprint. Durante el desarrollo del Sprint puede surgir dudas acerca del objetivo del Sprint por lo que el Scrum Master es el encargado de gestionar las dudas. Para gestionar las inquietudes sobre el objetivo del Sprint, el Scrum Master debe planificar los Daily Scrum. Los Daily Scrum debe realizar con Team Scrum, con el objetivo de solventar las dudas de todo los Developers tienen una duración de 15 minutos.

Cuando se termine el plazo de realizar el Sprint asignado a los Developers, él Scrum Master realiza una reunión con el fin de revisar las actividades realizadas y verificar si logro cumplir con el objetivo del Scrum. En la revisión están involucrados el Scrum Master, Developer y Product Owner. Se cierra el Sprint siempre y cuando se cumpla con los objetivos del Sprint, en caso de no cumplir con los objetivos se reasigna el Sprint hasta la próxima revisión. La revisión tiene una duración aproximada de 4 horas.

Para finalizar la fase del Juego se realiza una retrospectiva de los Sprint asignado, en este punto los Developers informan las dificultades que tuvieron al realizar el Sprint. La reunión para la retrospectiva tiene una duración de 4 horas.

Iteration, Milestone - Sprint

Las iteraciones son una manera de gestionar los Sprint de Scrum, cada iteración es un sprint. Las iteraciones pueden tener los estados de New, Backlog, Ready, In progress, In review, Done.

Figura 19
Iteraciones

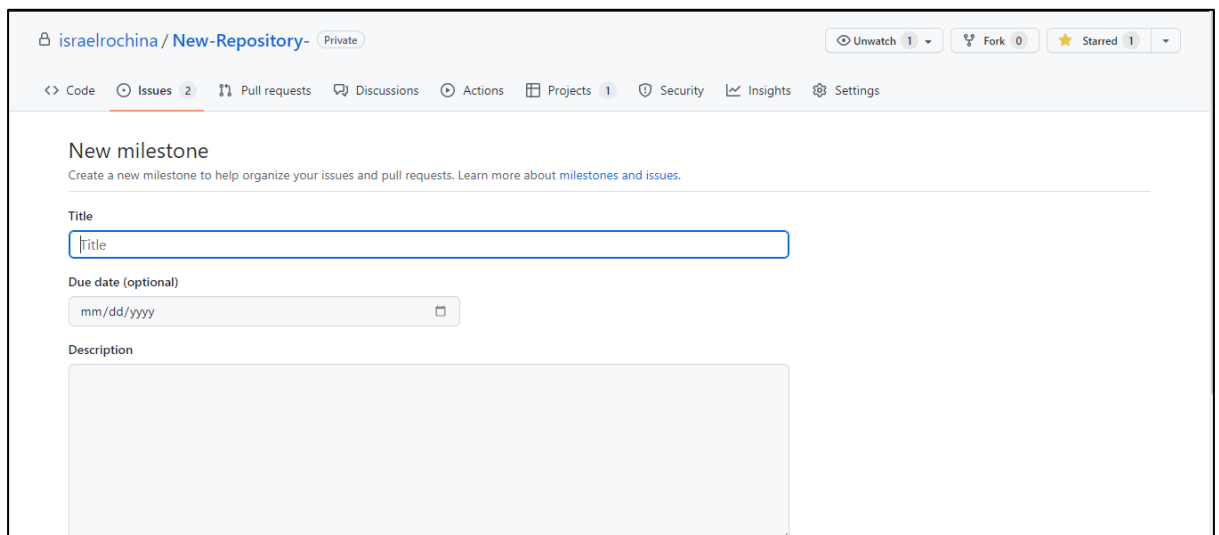


Dentro de GitHub también existen los Milestones, permite agrupar a las issues, se puede utilizar para organizar los Sprint Review o retrospectiva. Ver Figura 20.

Dentro de los Milestone se puede realizar varias acciones:

- Agrupar varios Issues.
- Incluir una descripción del objetivo que se desea lograr.
- Establecer una fecha de cierre.

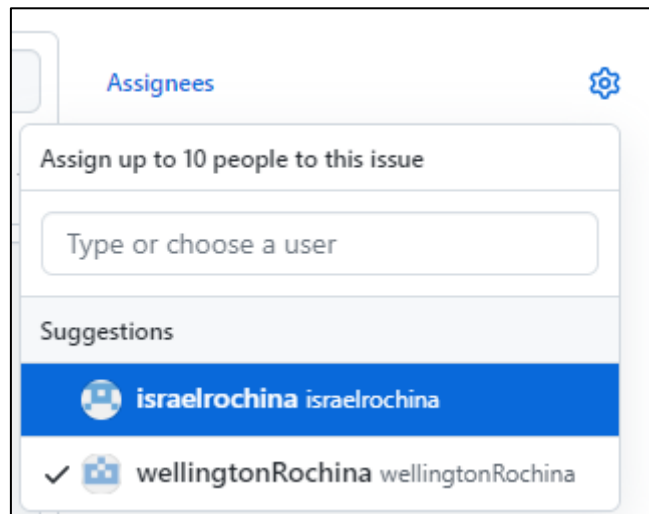
Figura 20
Milestone



Assignees

Como se puede observar en la Figura 21 dentro de la herramienta GitHub se puede asignar una Issue a un miembro del equipo, es la manera de asignar la tarea a uno o más miembros del equipo. El número máximo de colaboradores que se pueden asignar a una Issue es 10.

Figura 21
Assignees

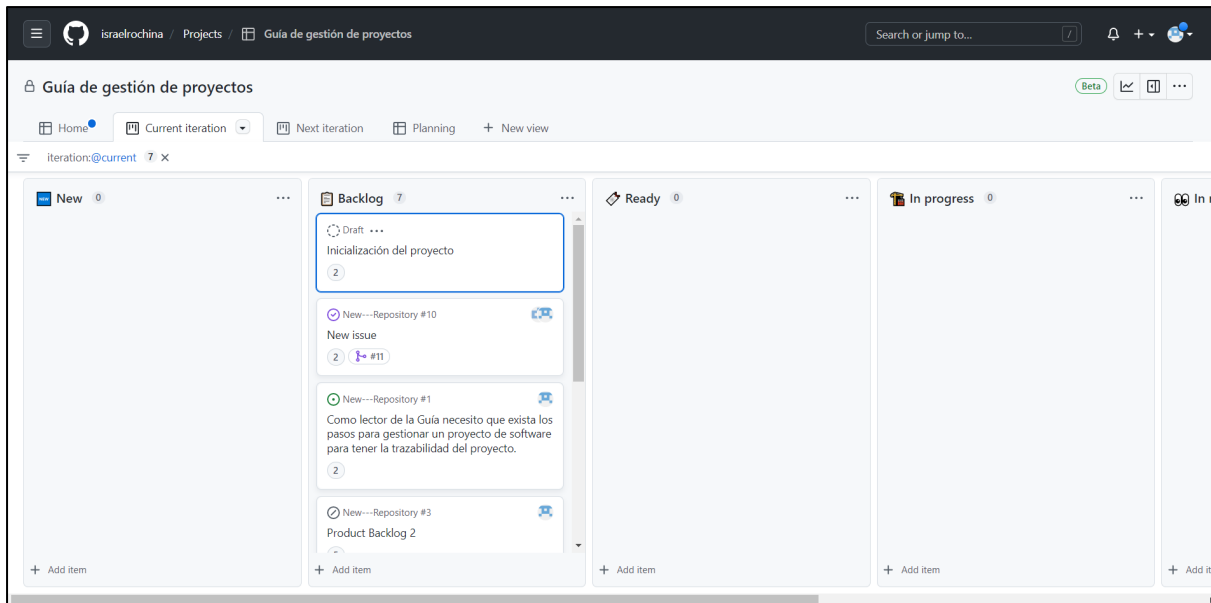


Projects – Gestionar Historias de usuarios

Como se observa en la

Figura 17 projects es una funcionalidad de GitHub el cual permite crear un tablero para la gestión de Issues, son personalizables, existen diferentes plantillas para su creación. Dentro de los projects se encuentran las iteraciones.

Figura 22
Projects

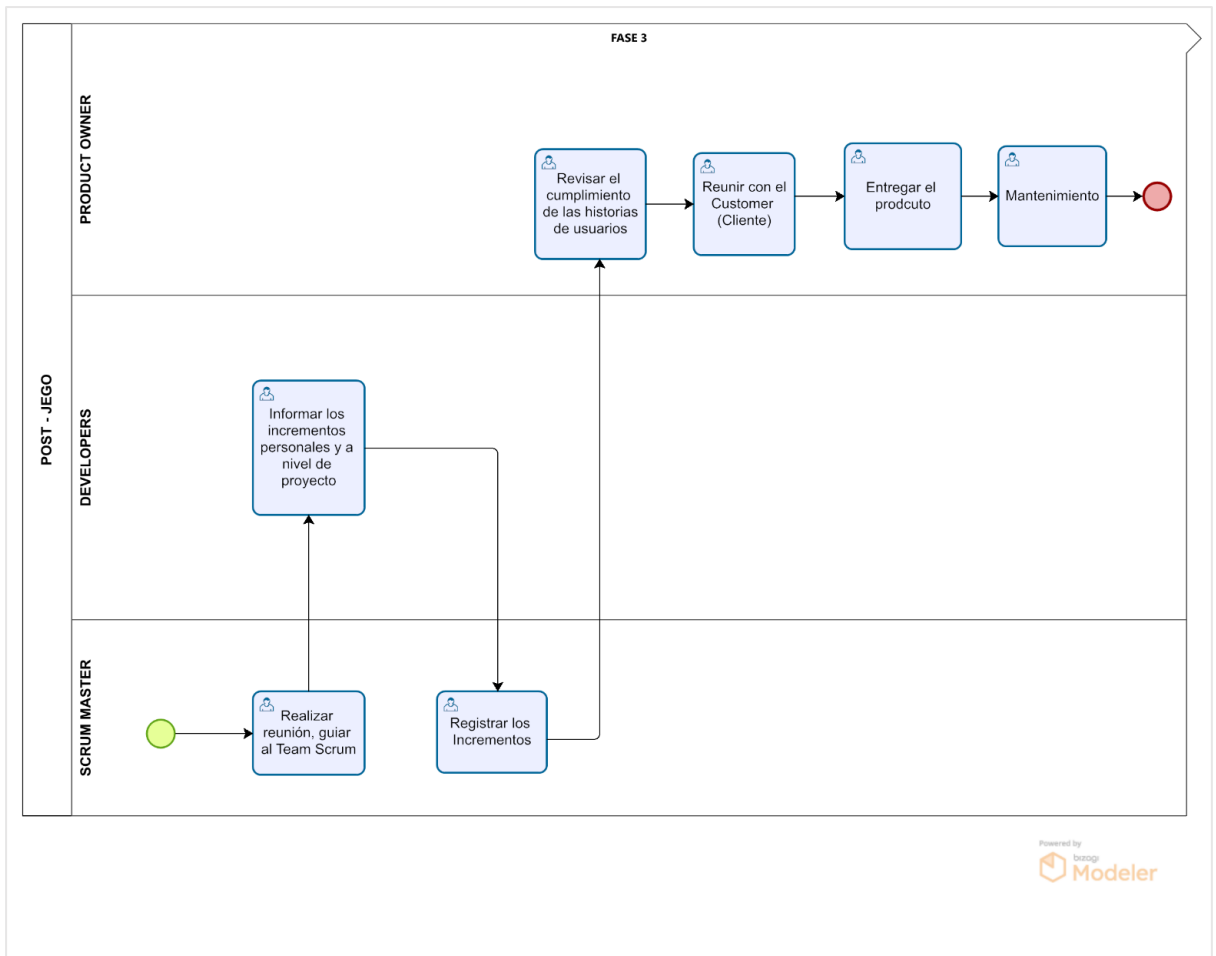


2.1.7. Post-Juego

La última fase de Scrum es el Post-Juego, en este punto se realiza una reunión con los Developers para saber cuáles son los incrementos. Se hace la entrega del producto al cliente, a futuro se puede realizar mantenimiento del producto.

Figura 23

Fase 3 Post-Juego



Luego de haber finalizado los Sprint es necesario realizar una reunión en el que está involucrado el Scrum Master y los Developers. La finalidad de la reunión es obtener la información de los incrementos que se ha obtenido al desarrollar el producto. También se analiza los incrementos a nivel personal.

Los incrementos del producto deben ser registrados por el Scrum Master. Los incrementos son revisados por el Product Owner juntamente con el Cliente con la finalidad de saber si se ha cumplido con los requerimientos y se ha cumplido con los objetivos planificados.

En caso de no estar satisfecho con los incrementos se regresa a la fase 1 y después realizar las demás fases hasta que se cumpla con los objetivos del proyecto. Si todos los incrementos están bien se procede a la entrega del producto.

A largo plazo, en caso de ocurrir algún problema con el producto entregado se procede a realizar el mantenimiento. El proceso del mantenimiento debe iniciar desde la fase 1 hasta concluir con la fase 3.

Tags

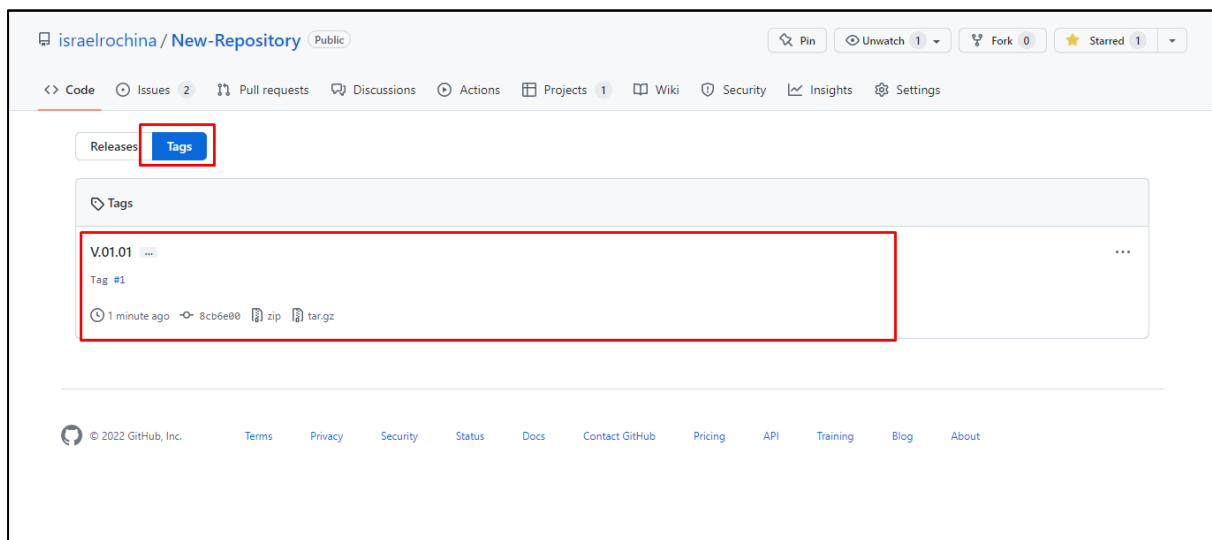
Entre las funcionalidades de GitHub existe la funcionalidad de crear tags. Los tags son utilizados para guardar un punto de recuperación. Para crear tags es recomendable que el producto de software esté funcionando de manera correcta. Los tags se deben cuando se termina los Sprint.

Las ventajas de crear tags:

- Los tags creados son respaldos no modificables.
- Se pueden recuperar información o código borrado durante el desarrollo.
- Los tags se pueden compartir en archivo comprimido y es portable.

Figura 24

Tags

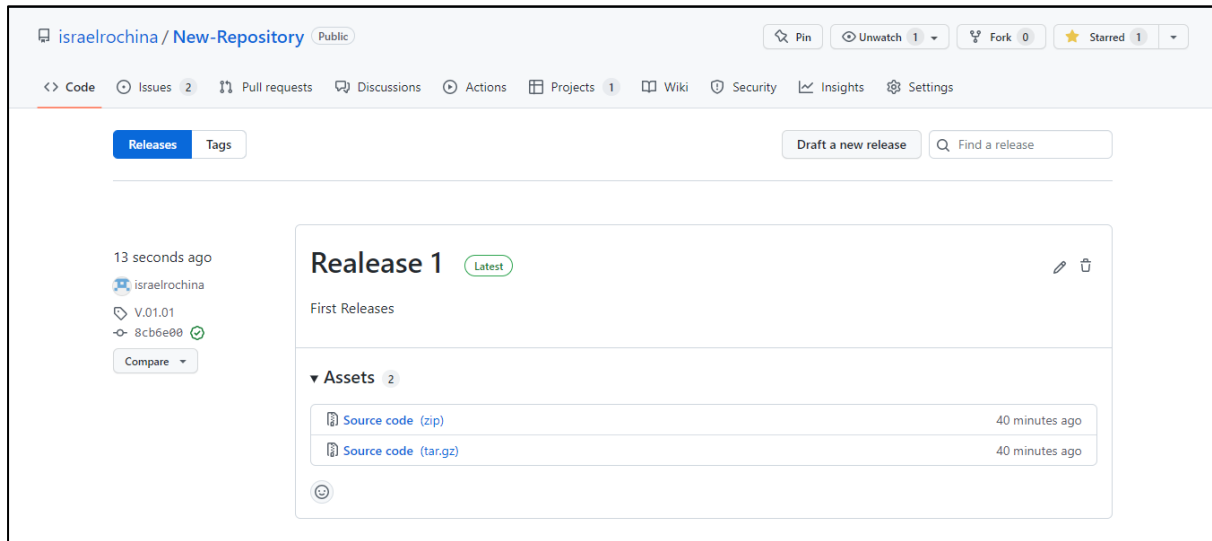


Releases

El Release permite almacenar una versión del proyecto que se está realizando, los releases están relacionados con los tags y son similares a los entregables de Scrum, es el producto funcional entregable.

Figura 25

Release



CAPÍTULO III

3. Desarrollo de la Guía metodológica

3.1. Introducción

La guía metodológica explica cómo gestionar proyectos relacionados a software, para la gestión se ha realizado una combinación del marco de trabajo Scrum y la herramienta GitHub. Por un lado, Scrum define los artefactos y eventos que se deben de emplear al desarrollar el software, por otro lado, las funcionalidades de la herramienta GitHub ayuda a gestionar los eventos y artefactos de Scrum.

Para fácil entendimiento la guía está estructurado por 3 fases propuesta en el marco de trabajo Scrum. Comenzando con el Pre-Juego, en el cual se planifica cuáles son las actividades para realizar dependiendo de la historia de usuario, se define los roles y se elabora el Product Backlog. Como segunda fase se tiene al Juego, en el que se desarrolla las historias de usuarios con una planificación de Sprint. Finalmente se tiene a la fase de Post-Juego, el cual consiste en realizar entregables o el producto final al cliente.

Cada funcionalidad que tiene GitHub es usada para gestionar las actividades realizadas dentro de las fases de Scrum. Dentro de la guía se ha incluido como iniciar un proyecto con GitHub, también se explica cada una de las funcionalidades de la herramienta. Para mejor entendimiento se incluyó gráficos.

3.2 Objetivo

Realizar una guía metodológica para la gestión de proyectos de software utilizando el marco de trabajo Scrum y gestionar las fases de Scrum con la herramienta GitHub.

3.3. Alcance

Desarrollar una guía metodológica para las personas que desean gestionar los proyectos de software. Gestionar las fases del marco de trabajo Scrum con las funcionalidades de la herramienta GitHub.

3.4 Audiencia

La guía se orientó para los desarrolladores, analistas, diseñadores todas las personas que están desarrollando o están inmersos en un proyecto de software.

3.5. Ejecución de las fases Scrum

3.5.1. Fase 1. Pre-Juego

Sprint 0

Primero, crear el proyecto con un nombre fácil de identificar o el nombre del problema a solucionar. Para tener todos los beneficios de la herramienta GitHub de manera gratuita, crear el proyecto en modo Público. En el apartado de descripción, incluir una breve descripción acerca de que se trata el proyecto detallado en la Figura 26.

Personalizar el estado de inicio de GitHub, se puede configurar el archivo **README.file** es un archivo de presentación al repositorio, se describe la problemática que se va a resolver o las tecnologías con cual se trabaja para el desarrollo de software.

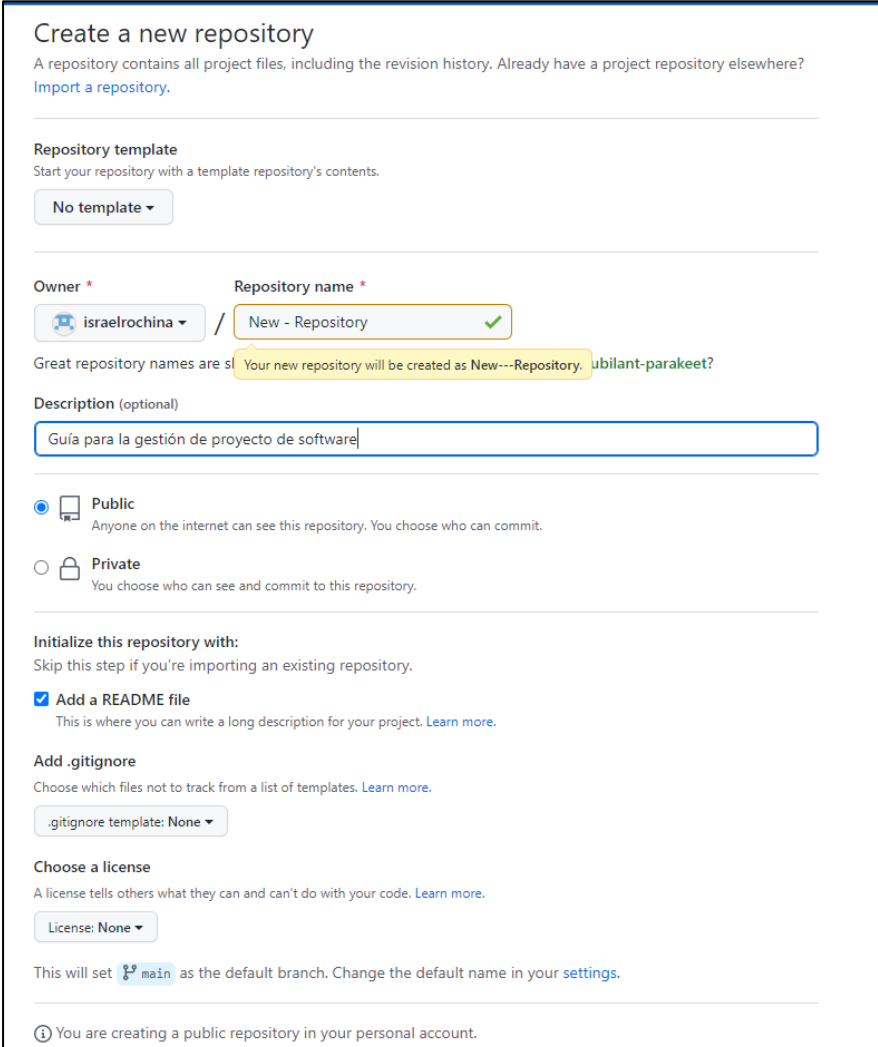
Por defecto la rama principal del repositorio tiene como nombre **main** o **master**, el nombre puede ser editado en caso de ser necesario.

Es importante tener en cuenta que para iniciar un proyecto es necesario tener una cuenta en la plataforma GitHub, la creación de la cuenta es fácil e intuitiva. Para crear una cuenta se lo realiza con una cuenta email y el ingreso de una clave con caracteres especiales y alfanuméricos, el número mínimo de caracteres es 8. Una vez creado la cuenta ya puede crear proyectos llamado en GitHub repositorio.

Tener en cuenta que los nombres de los repositorios son únicos, es decir que no puede existir dos repositorios con el mismo nombre. GitHub no hace distinción de letras mayúsculas o minúsculas en cuanto a nombre de repositorio.

Figura 26

Creación del Repositorio GitHub.



The screenshot shows the GitHub 'Create a new repository' page. At the top, it says 'Create a new repository' and provides a brief explanation of what a repository is. Below this, there is a 'Repository template' section with a 'No template' dropdown. The 'Owner' is set to 'israelrochina' and the 'Repository name' is 'New - Repository', which is highlighted with a green checkmark. A yellow tooltip points to the repository name, stating 'Your new repository will be created as New---Repository. ubilant-parakeet?'. The 'Description' field contains the text 'Guía para la gestión de proyecto de software'. The 'Public' option is selected, with a note that 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option is also visible. Under 'Initialize this repository with:', the 'Add a README file' checkbox is checked, with a note that 'This is where you can write a long description for your project. Learn more.' The '.gitignore' section has a dropdown set to 'None'. The 'Choose a license' section also has a dropdown set to 'None'. At the bottom, it states 'This will set main as the default branch. Change the default name in your settings.' and a note: 'You are creating a public repository in your personal account.'

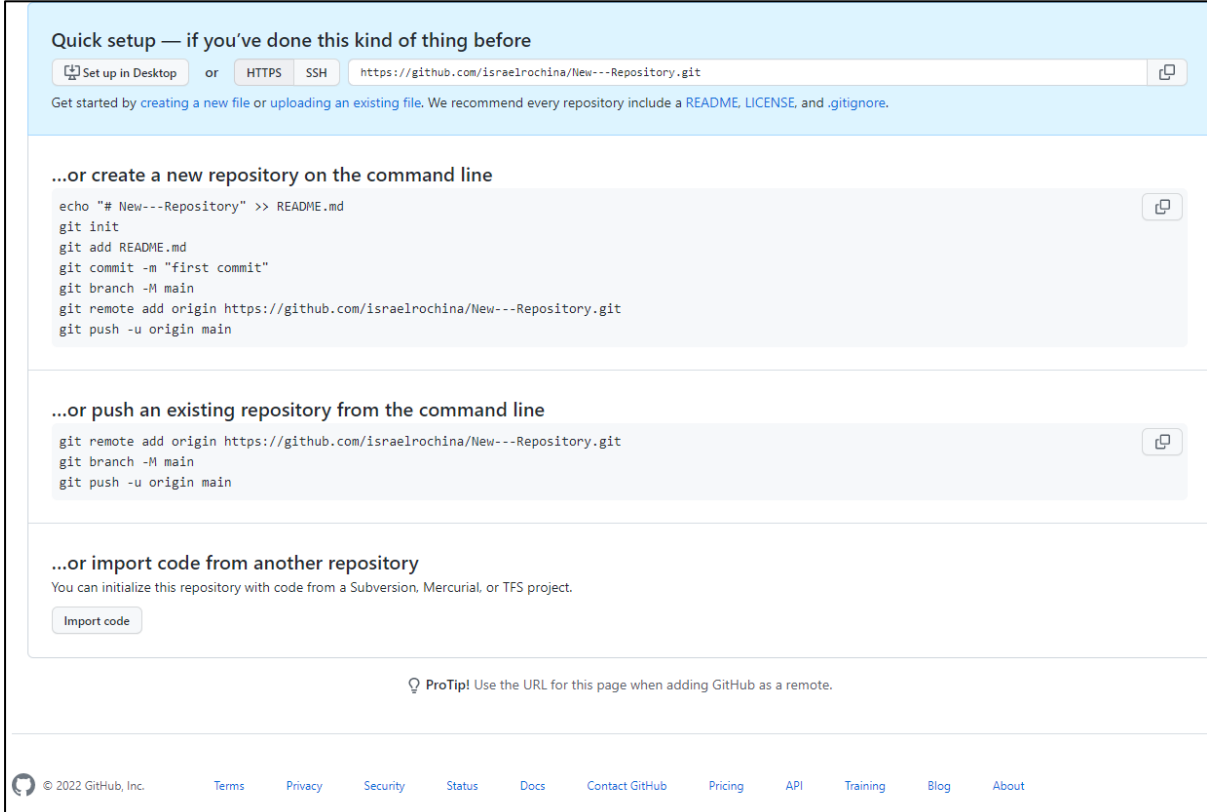
Inicializar el repositorio creado

Una vez creado el repositorio hay que inicializar el repositorio, para el cual GitHub cuenta con dos opciones. El primero es realizar una clonación del repositorio creado, para esto se debe utilizar de manera local Git. Con el comando `git clone` y la dirección HTTPS o SSH proporcionado por GitHub, luego se debe crear un archivo de preferencia un `README.md` posteriormente subir los cambios al repositorio remoto GitHub.

La otra manera es inicializar Git para un proyecto de manera local y utilizar las líneas de comandos que proporciona el repositorio Git. La Figura 27 contiene la línea de comandos para inicializar el repositorio.

Figura 27

Comandos para inicializar el repositorio GitHub



Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH `https://github.com/israelrochina/New---Repository.git`

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# New---Repository" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/israelrochina/New---Repository.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/israelrochina/New---Repository.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

ProTip! Use the URL for this page when adding GitHub as a remote.

© 2022 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

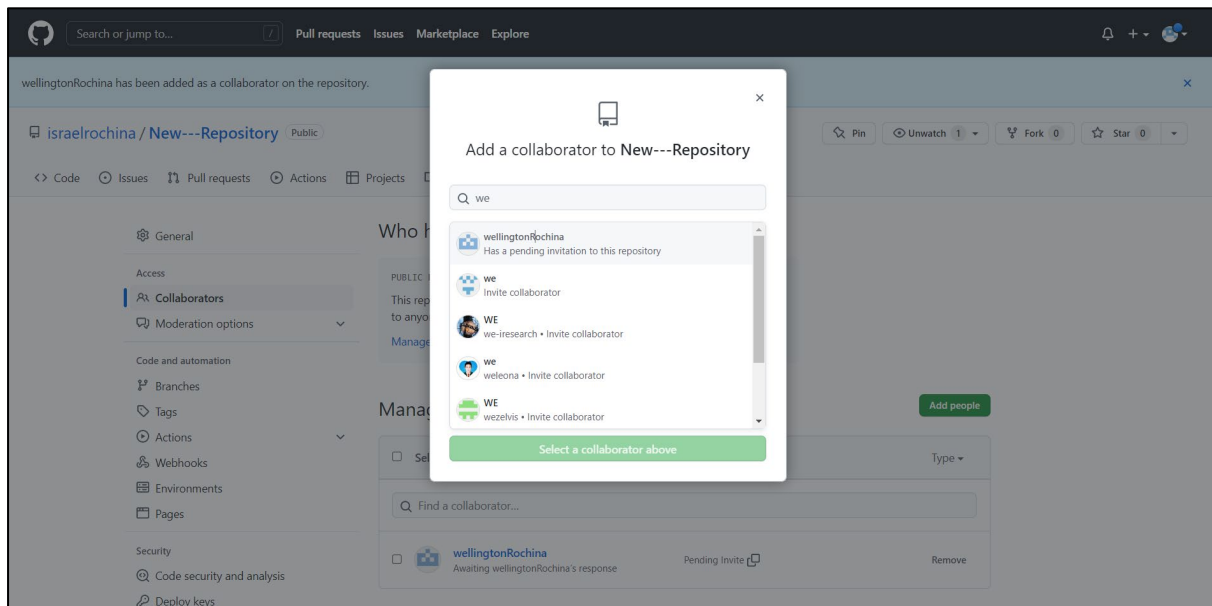
Agregar colaboradores, asignación de roles al proyecto

El paso fundamental para iniciar el marco de trabajo Scrum es la asignación de Roles, los roles que pueden asignar dentro del marco de trabajo Scrum son: Scrum Master, Product Owner y Developers.

Para agregar a los miembros del equipo al proyecto, dentro de las funcionalidades de GitHub. Situar en la ruta de Settings/Collaborators, dentro dar clic en **Add people**. Para agregar colaboradores ingrese el nombre completo o el email del colaborador como se muestra en la Figura 28.

Figura 28

Agregar colaboradores al proyecto.



Creación del Product Backlog

El producto Backlog parte de las historias de usuarios, por lo que se debe tener una reunión con Team Scrum. El Product Owner es el encargado de explicar cuáles son las necesidades del cliente, que funcionalidades se va a implementar, cual es el diseño del producto, las herramientas que se utilizará. El Scrum Team debe estar bien informado del objetivo central que desea cumplir el proyecto.

La estructura que debe tener un Product Backlog se presenta en la Tabla 10.

Tabla 10

Estructura del Product Backlog.

Estructura	Concepto
Identificador (Id) de la historia	Es un código único asignado para una historia de usuarios. Una vez asignado no debe reutilizarse ni si quiera cuando una historia se descarte. El código se usa para identificar o hacer referencias en otros documentos.
Enunciado de la historia, titulo	Es el nombre de la historia de usuarios. El formato para utilizar puede ser:

	Como Rol , necesito breve descripción de la funcionalidad que requiere , con la finalidad el resultado que espera obtener .
Estado	Permite identificar el estado en el que se encuentra la historia de usuario. Puede encontrarse en los siguientes estados: To do, In Progress, Done o Discard.
Dimensión/Esfuerzo	Es la medida de esfuerzo que necesita emplear para desarrollar la historia de usuario.
Iteración (Sprint)	Es la Iteración o Sprint en el que se encuentra la historia.
Prioridad	Es el valor de importancia que tiene la historia de usuario para el proyecto.
comentarios	Detalle, explicación de la historia de usuario.

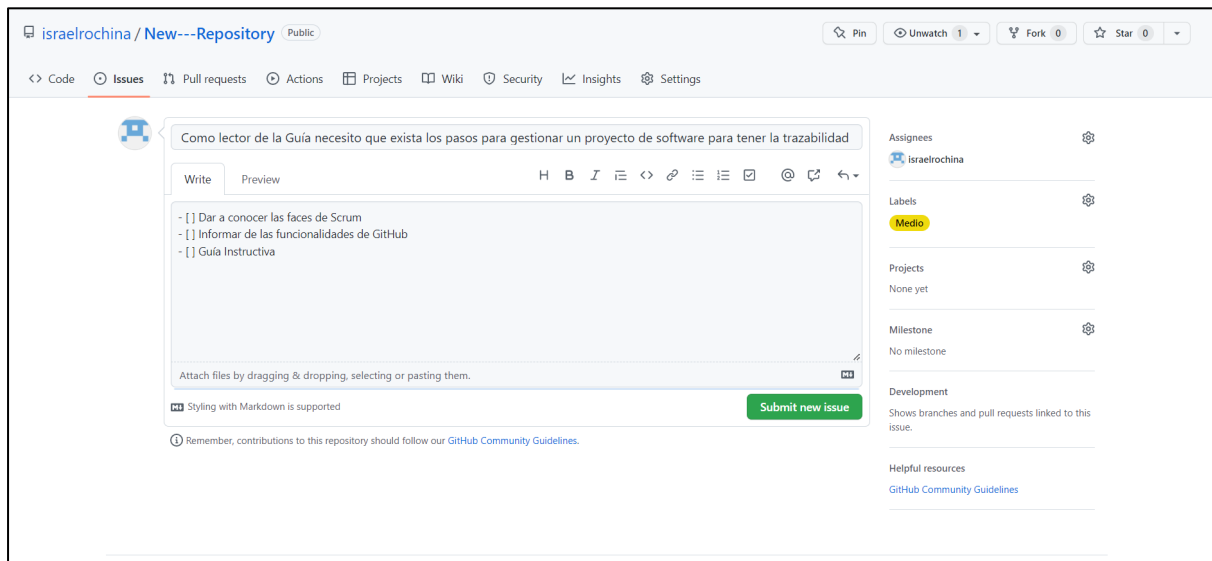
Issues

Las issues son semejantes al Product Backlog. Para crear una issue se debe ubicarse en el menú al apartado de **Issues** dentro dar clic en New issue.

Al momento de crear la Issue se debe ingresar los siguientes datos. Ver Figura 29.

- **Title:** Ingresar el nombre de la Issue con la estructura den enunciado de la historia ver Tabla 10
- **Comment:** Ingresar la descripción de la Issue, en este parte se lista las actividades que se debe realizar en Issue. [Clic aquí.](#)
- **Assigness:** Asignación al responsable de realizar la tarea.
- **Labels:** Son etiquetas para identificar el nivel de dificultad y la prioridad que tiene. Se puede personalizar y crear etiquetas. [clic aquí.](#)
- **Projects:** Es la forma de organizar los Product Backlogs, dentro de estos están las iteraciones equivalentes al Sprint de Scrum.
- **Milestone:** Los milestone son personalizables, permiten agrupar issues, [clic aquí.](#)

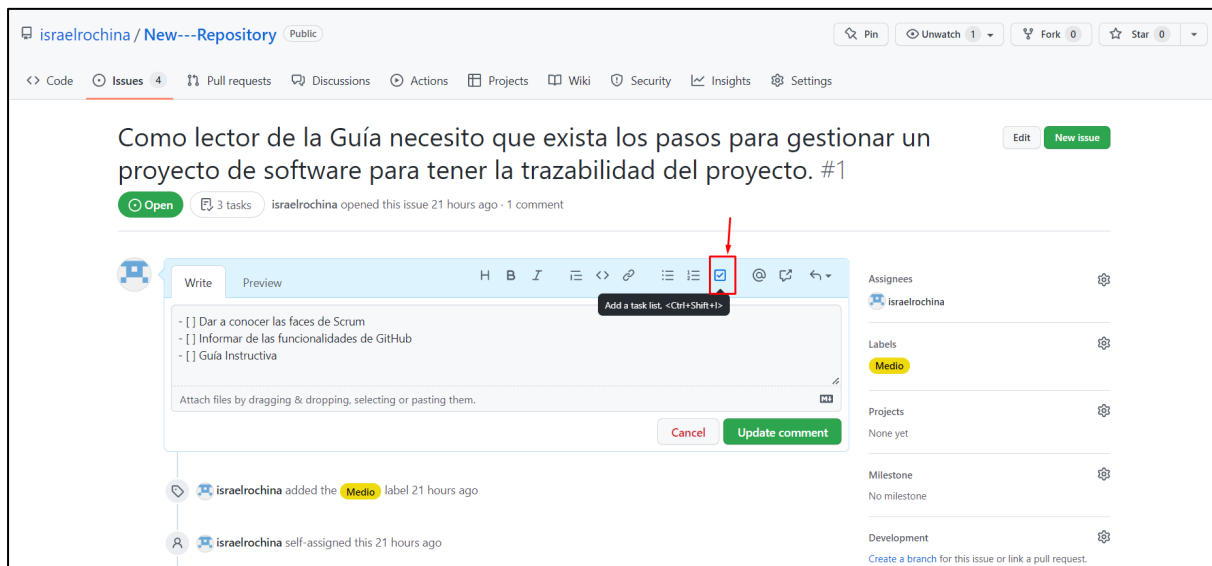
Figura 29
Creación de Issue.



Lista de tareas

Para realizar una lista de tareas dentro de la Issue, dar clic dentro de la Issue en el apartado de comentarios se encuentra la opción de add task list. Ver Figura 30.

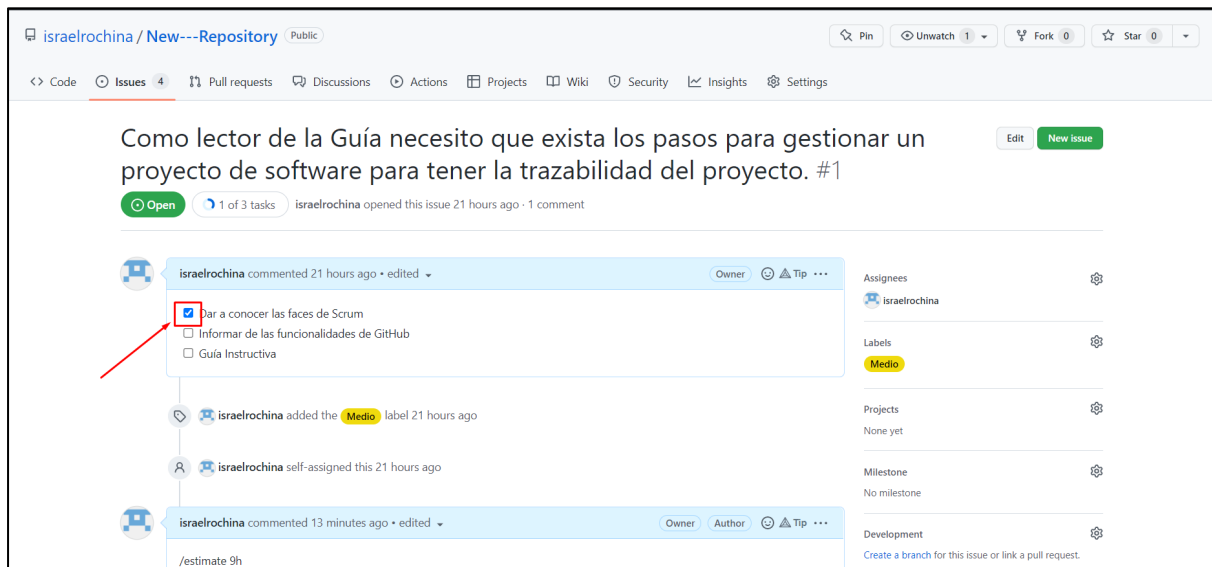
Figura 30
Añadir tarea a la issue.



Como se observa en la Figura 31 las tareas que realice tendrán un check de haber concluido con la tarea, para marcar como finalizado dar clic sobre la tarea.

Figura 31

Completar una actividad en la Issue.



Creación de Labels

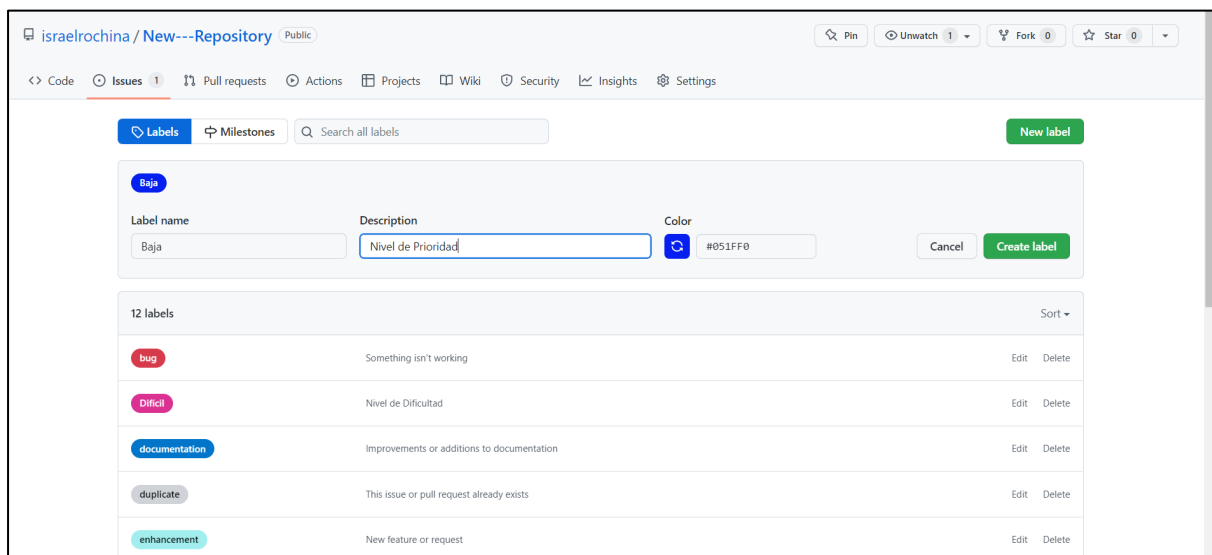
Para crear Labels personalizados, ubicarse en el menú apartado Issues dentro dar clic en Labels. Una vez dentro dar clic en New Labels.

Para crear los Labels ingresar los siguientes campos. Ver

Figura 32.

- **Label name:** Ingresar el nombre del Label.
- **Description:** Ingresar un detalle de para qué es el Label.
- **Color:** Seleccionar un color.

Figura 32
Crear Label.



3.5.2. Fase 2. Juego

Creación de Sprint Planning

El Scrum Máster debe crear y dirigir la reunión a esto se llamará Sprint Planning.

Para gestionar el Sprint Planning se hará uso de la Discussion, esto permite generar un hilo de conversación y registro de información. Para crear un Discussion dirigirse al apartado de menú en Discussion dentro dar clic en crear Discussion. Ver

Figura 33. Tener en cuenta que debe estar activado Discussion en las configuraciones de GitHub en el apartado de setting en la parte final. Ver Figura 34.

Para crear el Discussion ingresar los datos en los siguientes campos.

- **Category:** Seleccionar el tipo de la discusión.
- **Title:** Ingresar el nombre de la discusión.
- **Description:** Ingresar la pregunta, empezar la conversación o hacer un anuncio.
- **Label:** etiqueta para identificar la discusión. Puede personalizar el Label, [clic aquí](#).

Figura 33

Creación de discusión.

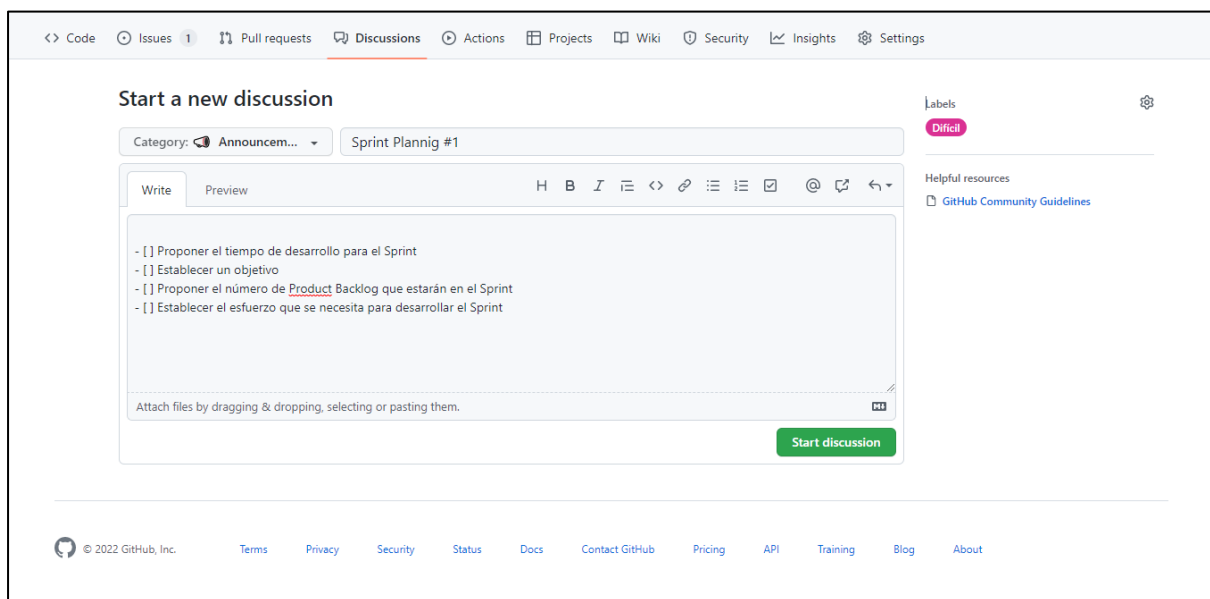


Figura 34

Activar discusión.

Discussions
Discussions is the space for your community to have conversations, ask questions and post answers without opening issues.

Get started with Discussions

Engage your community by having discussions right in your repository, where your community already lives

[Set up discussions](#)

Crear Milestone para organizar las issues

Para organizar y agrupar las issues, se debe crear los milestones. Para crear dar clic en New milestone.

Ingresar los siguientes datos, ver Figura 35.

- **Title:** Ingresar el nombre.
- **Due date:** Ingresar la fecha cierre.
- **Description:** Ingresar el objetivo o el detalle.

Figura 35

Creación de milestone.

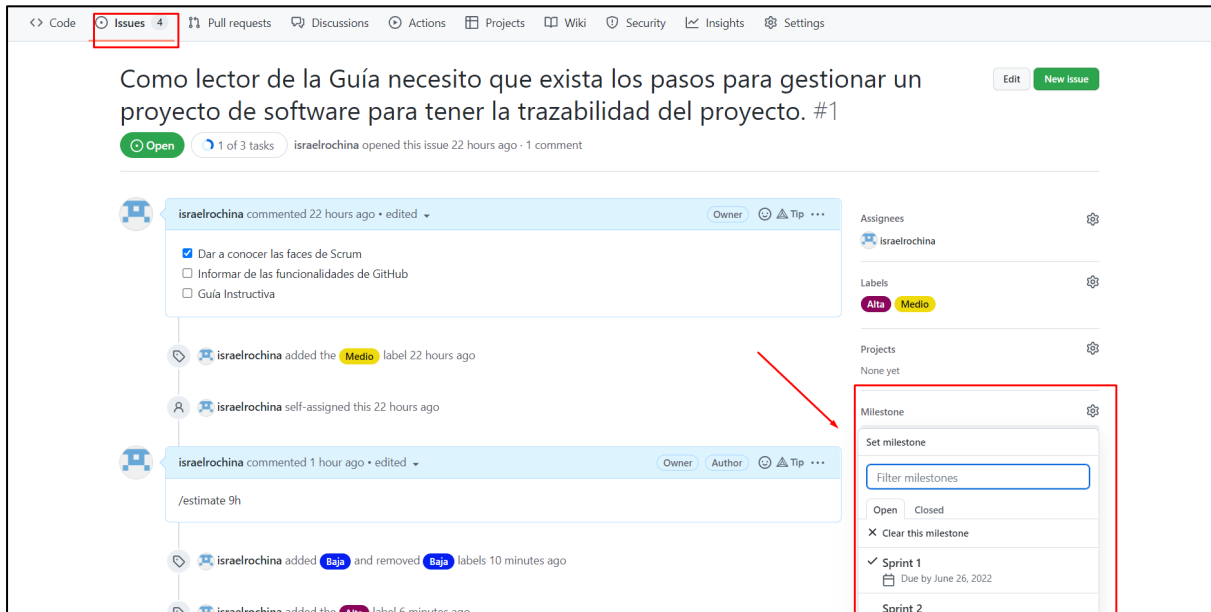
The screenshot shows the 'New milestone' form in a GitHub repository. The navigation bar at the top includes 'Code', 'Issues 4', 'Pull requests', 'Discussions', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The form fields are: 'Title' with the value 'Sprint 1'; 'Due date (optional)' with the value '06/26/2022'; and 'Description' with the text 'Realizar el instructivo de la Guía Metodológica, incluir capturas.'. A green 'Create milestone' button is located at the bottom right of the form. The footer of the page contains copyright information and various links like 'Terms', 'Privacy', 'Security', 'Status', 'Docs', 'Contact GitHub', 'Pricing', 'API', 'Training', 'Blog', and 'About'.

Asignar o relacionar un Product Backlog (Issue) al milestone

Como se observa en la Figura 36 para asignar una Issue al milestone, dirjase al menú en el apartado de Issue. seleccione la Issue que desea asignar al milestone. En las configuraciones seleccione el milestone deseado.

Figura 36

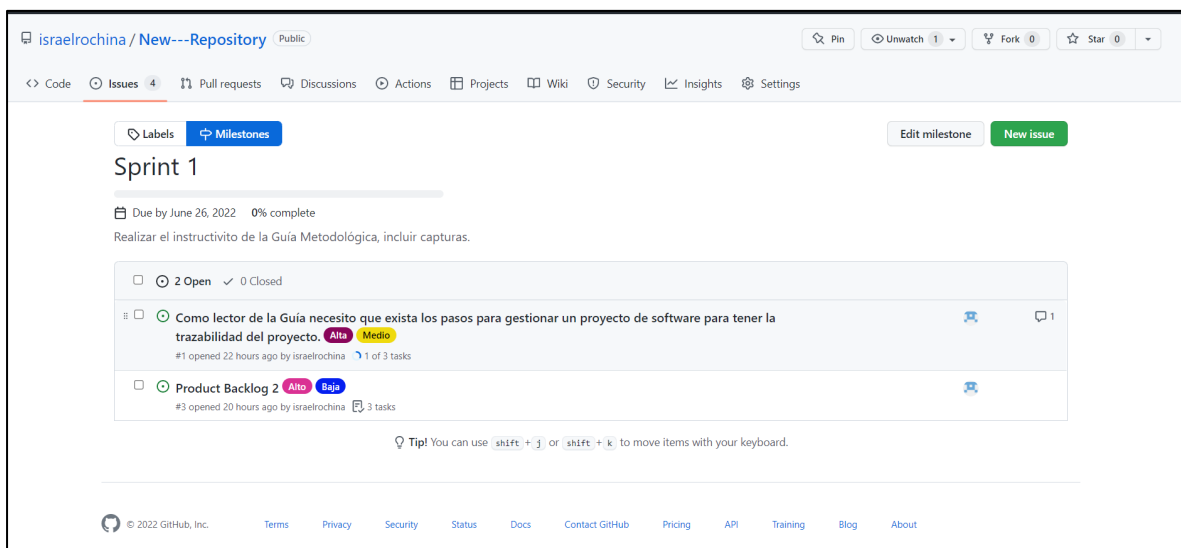
Asignar Issue a Milestone.



Para visualizar los Product Backlog que están asociados al milestone. Diríjase al milestone y habrá el milestone. Dentro puede ver la lista de Product Backlog asociados, también puede ver la barra de porcentajes de las Issue Finalizadas. Ver Figura 37.

Figura 37

Lista de issues en milestone.



Creación y gestión de Sprint

La herramienta GitHub proporciona la opción de crear Proyectos (Project). Los Project permiten realizar un seguimiento a las Issues. Par crear un Project ubicarse la barra del menú

del repositorio y hacer clic en **add Project** y selección “Ir al perfil para crear un nuevo proyecto”. Ver Figura 38.

Dentro del perfil dar clic en nuevo proyecto, seleccione la plantilla **feature** (La plantilla permite organizar por iteraciones las cuales son el Sprint, también permite visualizar en modo Tabla o tarjeta). Ver Figura 39.

Figura 38

Ingresar al perfil para crear un proyecto.

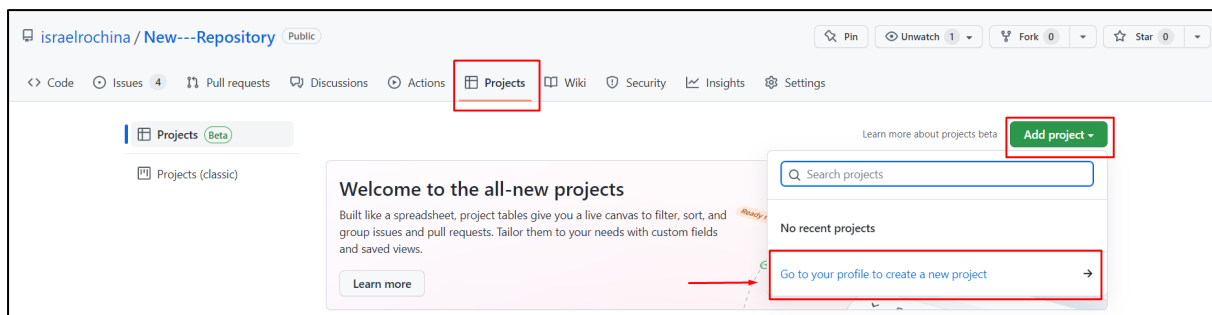
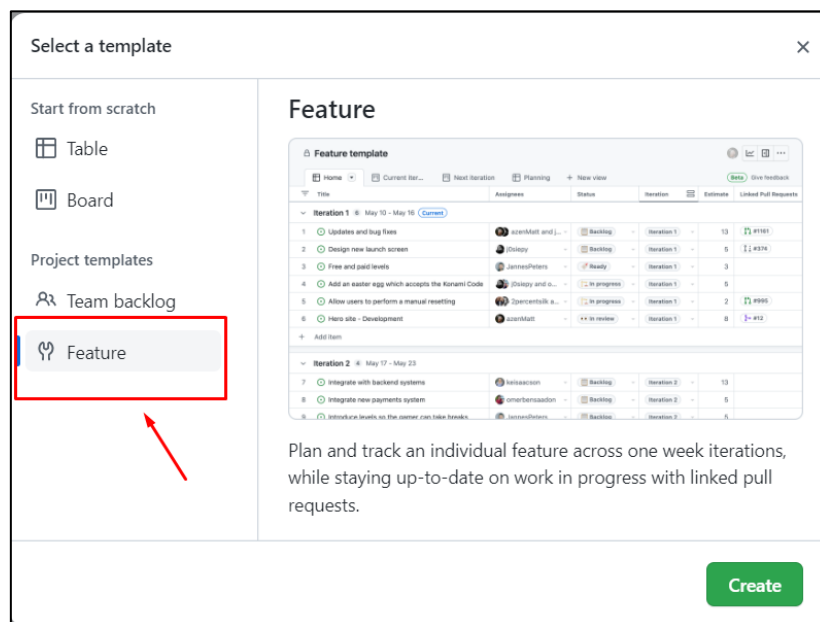


Figura 39

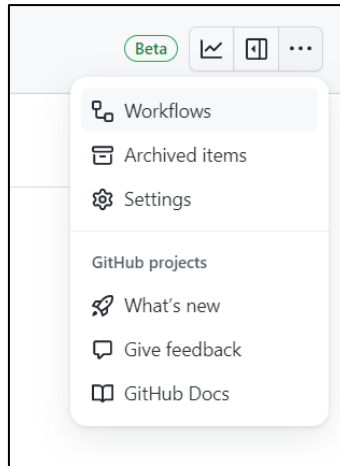
Seleccionar la plantilla para el proyecto



El proyecto es personalizable. Para configurar el proyecto, dar clic en los tres puntos ubicados en la parte izquierda superior, continuación seleccionar **settings**. Ver Figura 40.

Figura 40

Configurar el proyecto.

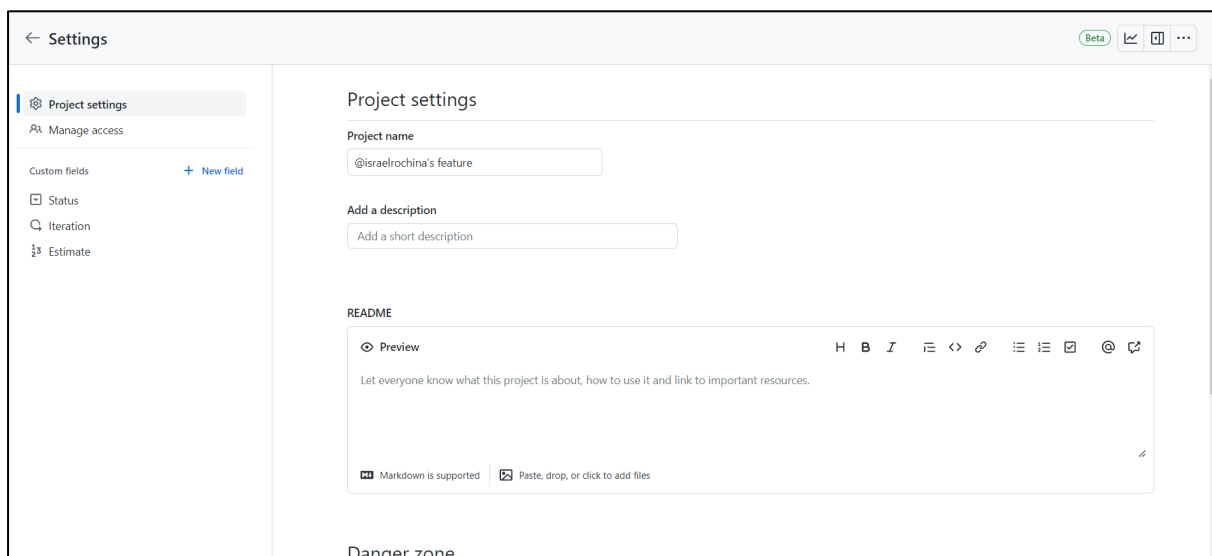


Dentro de las configuraciones debe hacer las siguientes configuraciones. Ver Figura 41:

- Configurar el nombre del proyecto
- Añadir una descripción
- Agregar un archivo README de presentación
- Administrar los accesos al proyecto
- Personalizar los campos para el proyecto

Figura 41

Opciones de configuración del proyecto



Estructura de Sprint Backlog

Una vez construida el Product Backlog se procede a la creación de los Sprint. Los Sprint aglomeran varios Product Backlog.

La estructura que debe tener un Sprint Backlog se presenta en la Tabla 11.

Tabla 11

Estructura de Sprint Backlog.

Estructura	Concepto
Identificador	Es un id único que permite identificar al Sprint.
Identificador de los Product Backlog	Es la lista de Id de los Product Backlog que están involucrados en la Sprint.
Objetivo	EL objetivo que desea cumplir al terminar el Sprint.
Nombre	Nombre del Sprint al que pertenece
Tiempo	Es el tiempo de la duración de un Sprint

Configurar número de iteraciones (Sprint)

Las iteraciones de GitHub son semejantes a los Sprint de Scrum. Para modificar los sprint creados por defecto en el proyecto, ir a configuraciones y seleccionar **Iteration**. Continuación eliminar las iteraciones haciendo clic en el icono



Una vez eliminado todas las iteraciones, seleccionar más opciones. Seleccionar la fecha de inicio de las iteraciones, la duración de las iteraciones en semanas o días. Continuación dar clic en **añadir**. Ver Figura 42. Cuando se necesite crear más iteración dar clic añadir iteraciones y guardar los cambios. Ver Figura 43.

Figura 42

Configurar iteraciones en el proyecto

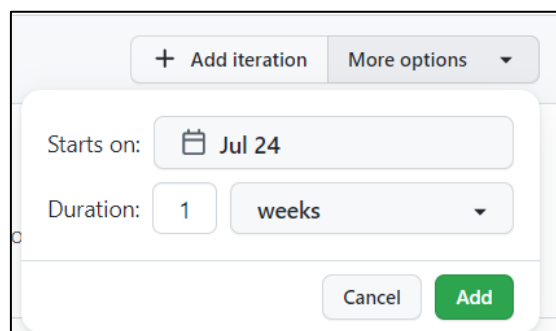
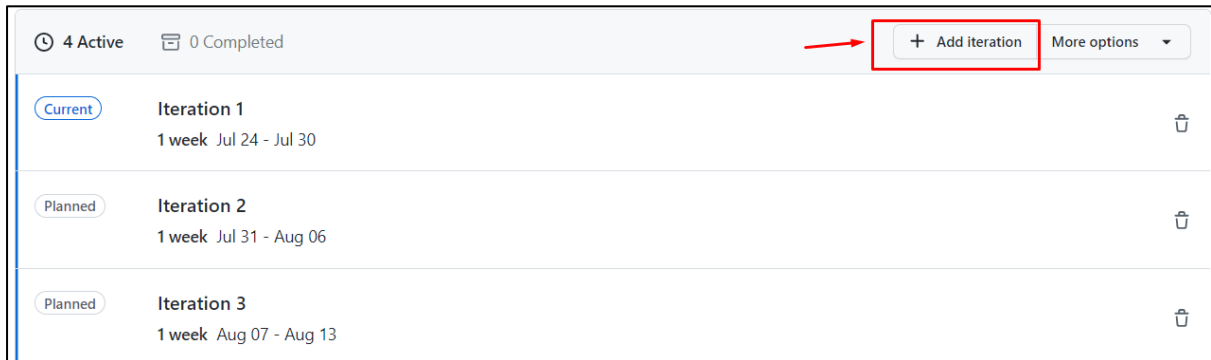


Figura 43

Añadir más iteraciones.

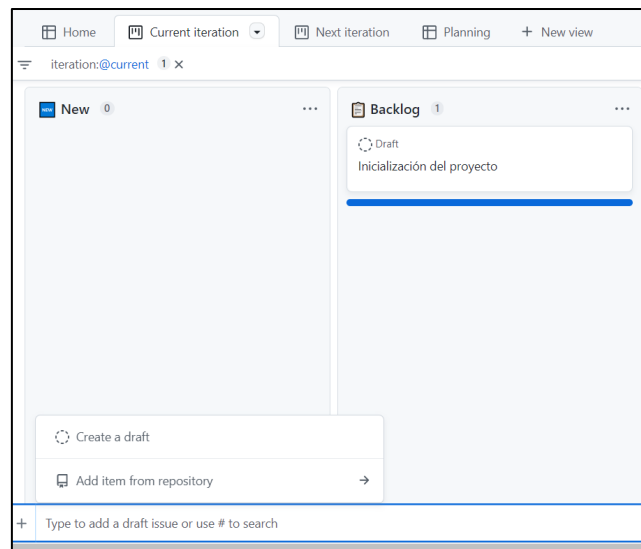


Agregar el Product Backlog (Issues) a las iteraciones (Sprint)

Para agregar las issues al proyecto, dentro del proyecto existe la vista **Current Iteration** (Iteración actual). Seleccione add item dentro de Backlog, tiene dos opciones para añadir un issue; agregar una issue existente en un repositorio o crear un draft y luego convertir en issue. Ver Figura 44.

Figura 44

Agregar Issue.

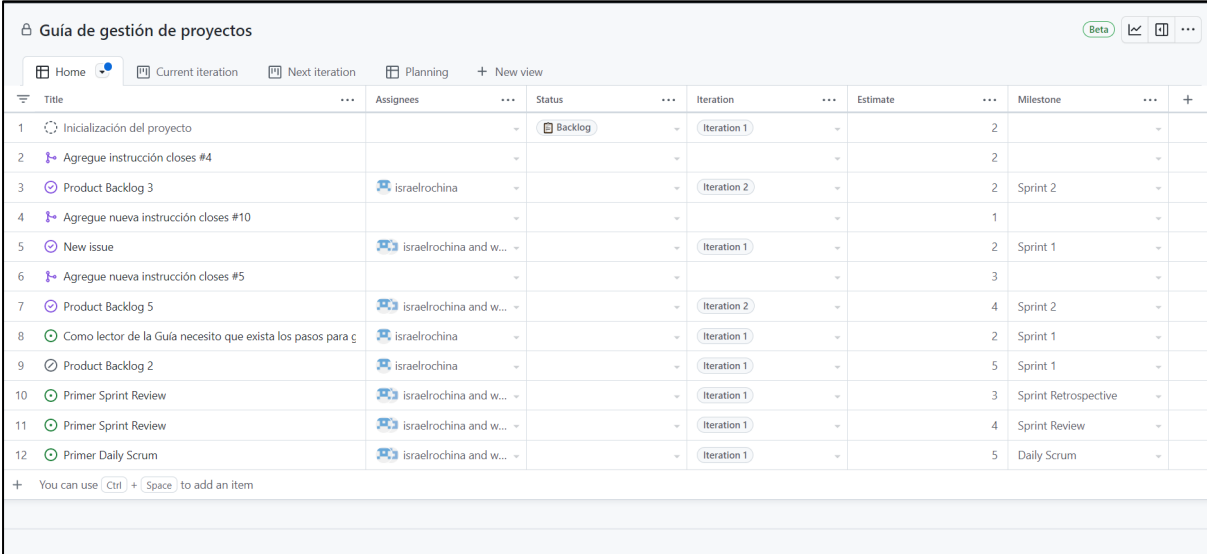


Asignar las Issue o Draft a las iteraciones (Sprint)

Cada Issue debe pertenecer a una iteración (Sprint). Para asignar las iteraciones a las issue, ubicarse en **Home** del proyecto, luego en la columna de **iteration** seleccionar la iteración. También puede asignar el tiempo estimado para realizar la issue. Como se observar en la Figura 45.

Figura 45

Asignación de iteraciones a las issues.



Title	Assignees	Status	Iteration	Estimate	Milestone
1 Inicialización del proyecto		Backlog	Iteration 1	2	
2 Agregue instrucción closes #4				2	
3 Product Backlog 3	israelrochina		Iteration 2	2	Sprint 2
4 Agregue nueva instrucción closes #10				1	
5 New issue	israelrochina and w...		Iteration 1	2	Sprint 1
6 Agregue nueva instrucción closes #5				3	
7 Product Backlog 5	israelrochina and w...		Iteration 2	4	Sprint 2
8 Como lector de la Guía necesito que exista los pasos para c	israelrochina		Iteration 1	2	Sprint 1
9 Product Backlog 2	israelrochina		Iteration 1	5	Sprint 1
10 Primer Sprint Review	israelrochina and w...		Iteration 1	3	Sprint Retrospective
11 Primer Sprint Review	israelrochina and w...		Iteration 1	4	Sprint Review
12 Primer Daily Scrum	israelrochina and w...		Iteration 1	5	Daily Scrum

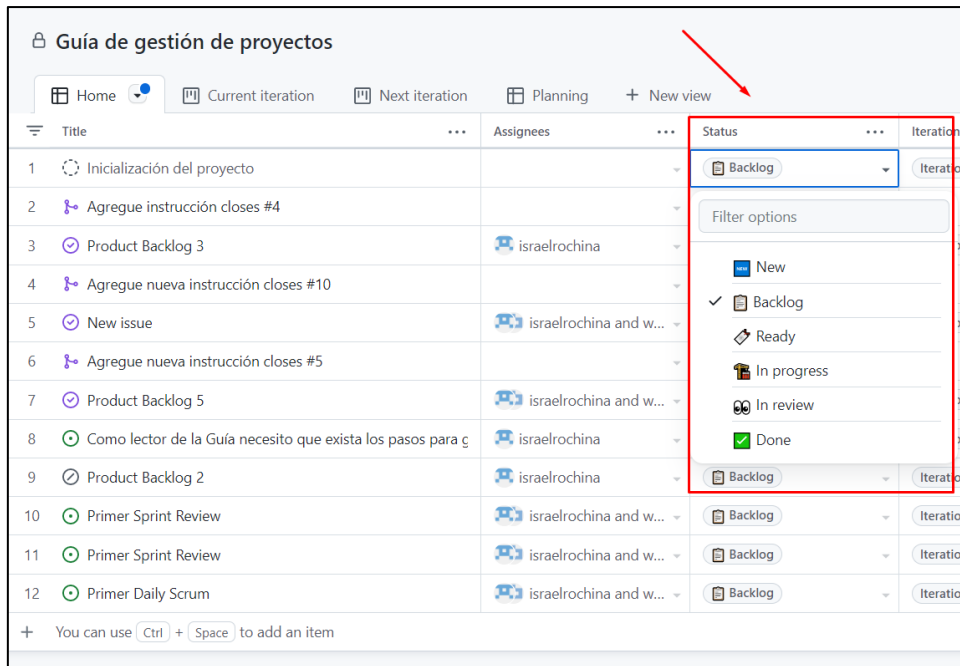
Asignación de Status a las Issues

GitHub denomina Status al estado en que se encuentra la issue, existen 6 estados incluidos en la plantilla feature como se observa en la Figura 46. Para seleccionar el estado ubicarse en Home del proyecto y en la columna status asignar cada estado a las issue.

- **New:** Estado cuando se está revisando que actividades tendrá la issue.
- **Backlog:** Estado cuando la issue está en espera de iniciar o ser asignado un developers.
- **Ready:** Estado cuando la issue está listo para iniciar a desarrollarse.
- **In progress:** Estado cuando la issue se está desarrollando.
- **In review:** Estado en el que el scrum master está revisando si la issue cumple con el objetivo.
- **Done:** Estado cuando una Issue ha cumplido con el objetivo.

Figura 46

Estados en el proyecto.



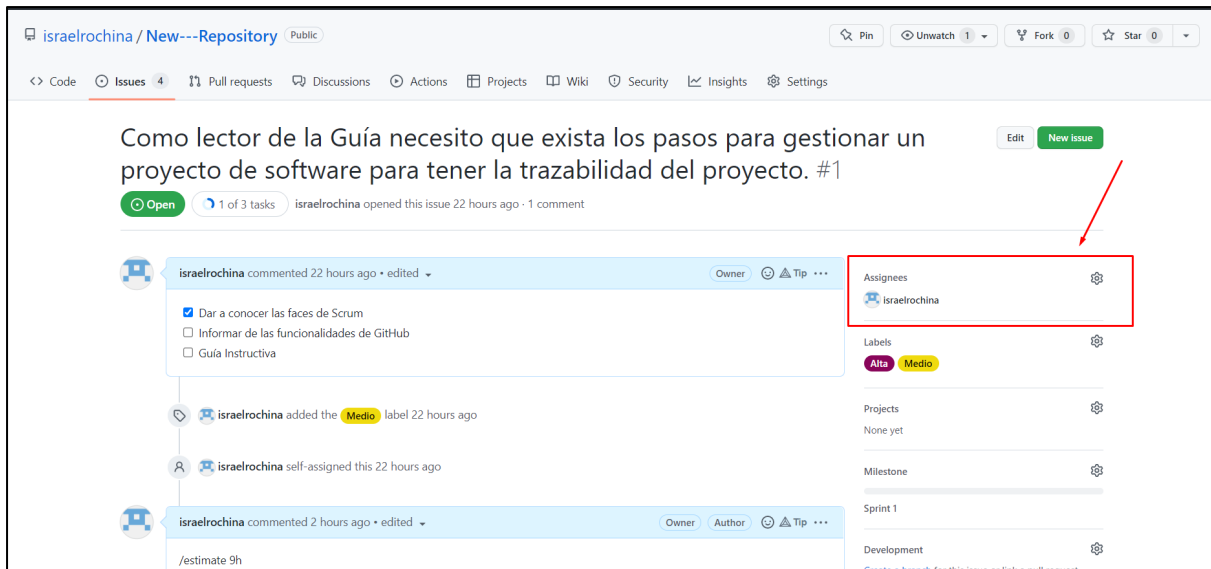
Ejecución de la iteración

Asignar tareas

Asignar las tareas o issues a cada developer. La asignación se puede realizar desde varias partes una forma de realizar es desde la misma Issue (Product Backlog), Dentro de las configuraciones de Issue en **assignees** seleccionar el responsable de la tarea. Ver Figura 47.

Figura 47

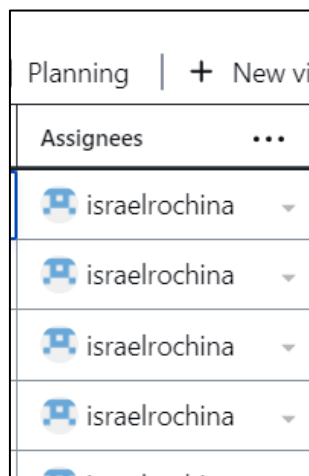
Asignar tarea.



Otra forma de agregar los developers que van a estar a cargo de la issue es desde el mismo tablero del Project. En la columna assignees se añade los responsables.

Figura 48

Asignar responsables de las issues desde el Project.



Ejecución de la issues por parte del developers

Una vez que se haya asignado la tarea o issue a un developer. Él responsable de la issue debe crear una rama por cada issue que se le haya asignado durante la primera iteración (Sprint). Para crear una rama ingresar a una Issue en el apartado de configuraciones dar clic en **Create a Branch**. Ver Figura 49. Acto seguido, seleccionar la rama que contine el código a desarrollar y dar clic en create branch. Ver Figura 50.

Figura 49

Crear una rama por un Product Backlog.

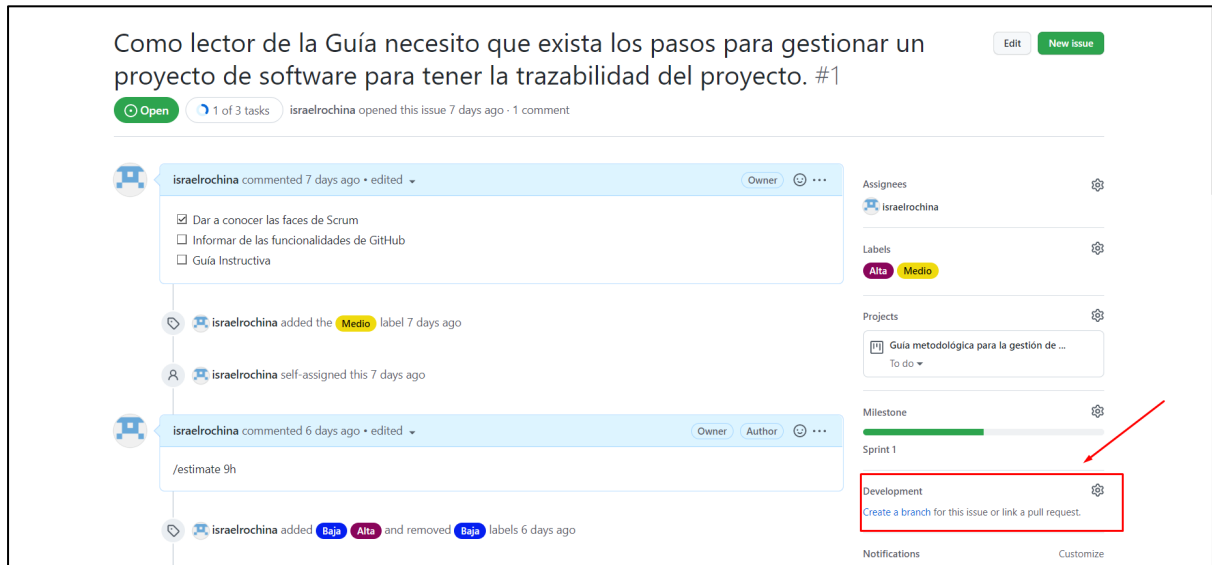
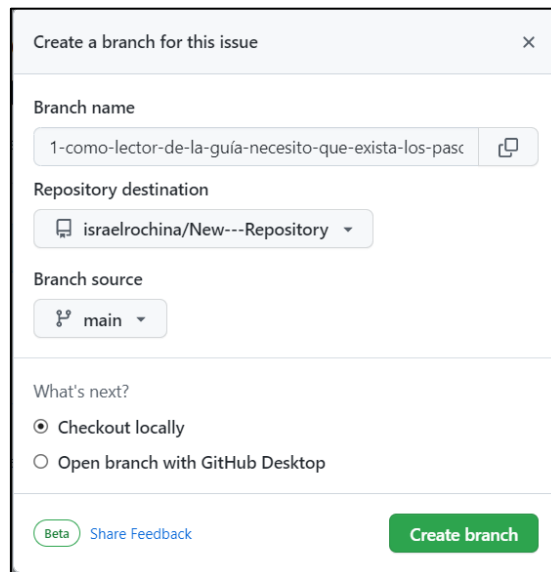


Figura 50

Configuración para crear una rama nueva.



Planificar Daily Scrum

Crear el Daily Scrum como una Issue y asignar a un Milestone llamado Daily Scrum con el fin de organiza las reuniones. Ver Figura 51 y Figura 52. El proceso de crear las Issue para la revisión debe ser a diario con una duración aproximada de 15 minutos para todo el equipo scrum que este participando en la iteración N (Sprint), debe estar en incluido en la iteración en el apartado Ready. Ver Figura 53.

Figura 51

Sprint Daily Scrum.

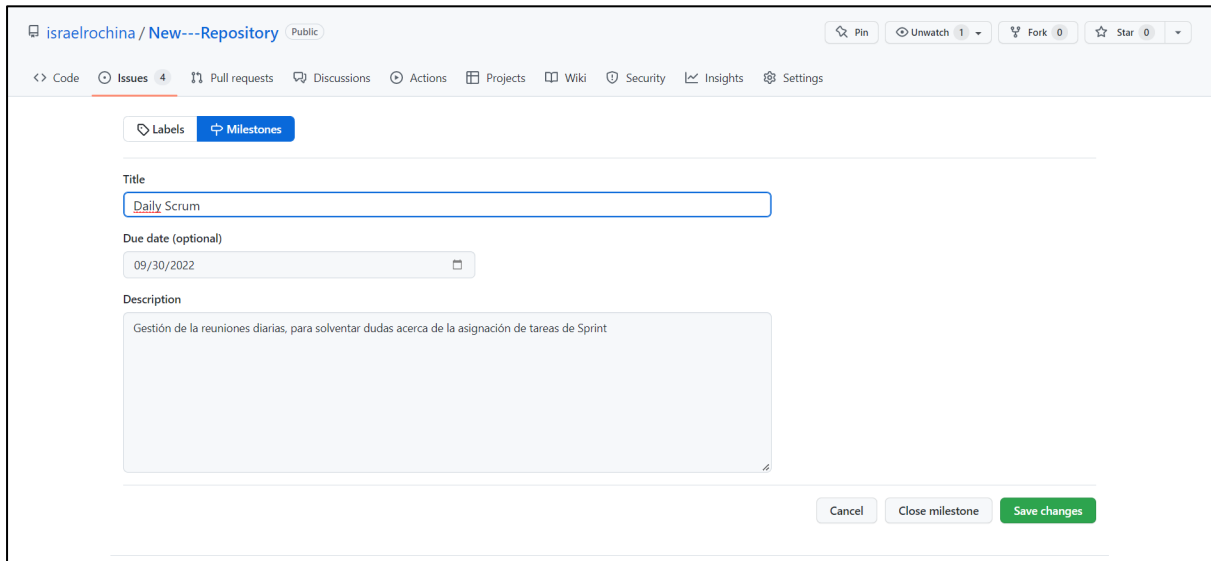


Figura 52

Creación Issue Daily Scrum.

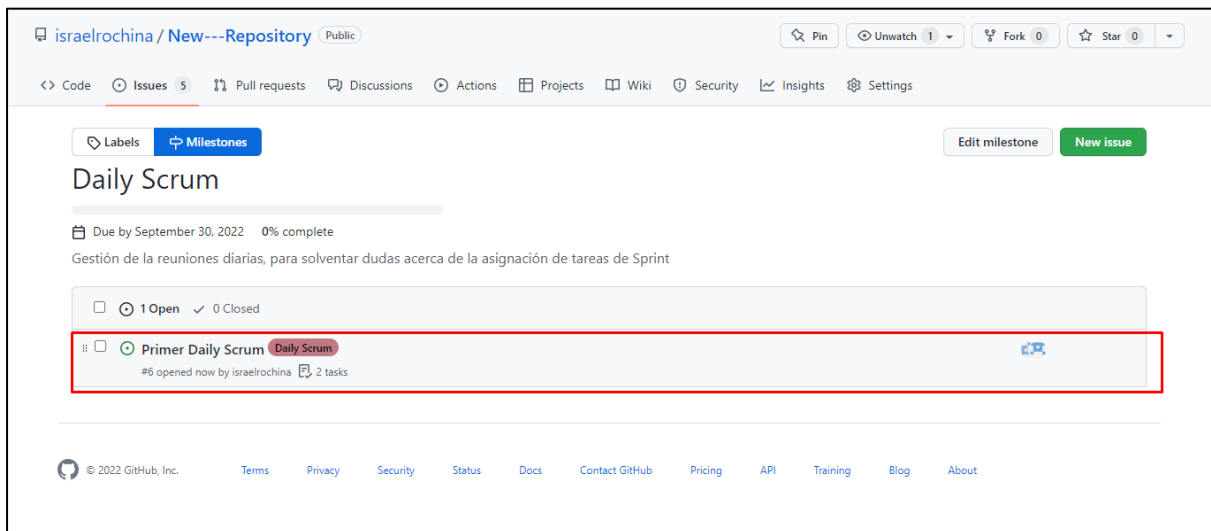
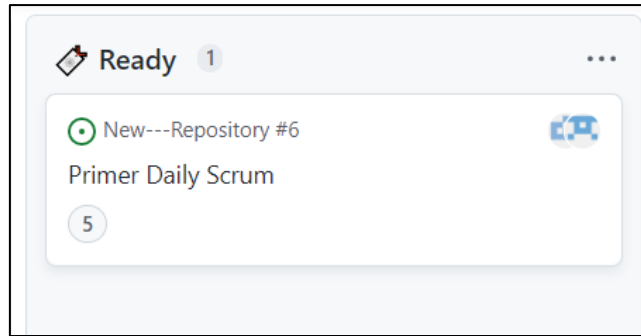


Figura 53

Daily scrum en la iteración.



Realizar Sprint Review

Una vez terminado un Sprint, antes de declarar como terminado realizar un Sprint Review, para analizar y saber si se cumplió con el objetivo del Sprint.

Crear un Milestone con el nombre Sprint Review con fecha de fin de proyecto. En la iteración (Sprint) añadir una Issue con el nombre Sprint Review 1,2, N. depende cuanta Sprint Review se realiza hasta finalizar el proyecto. Ver Figura 54. El Sprint review debe ser asignado a cada iteración. Ver Figura 55.

Figura 54

Creación milestone Sprint Review

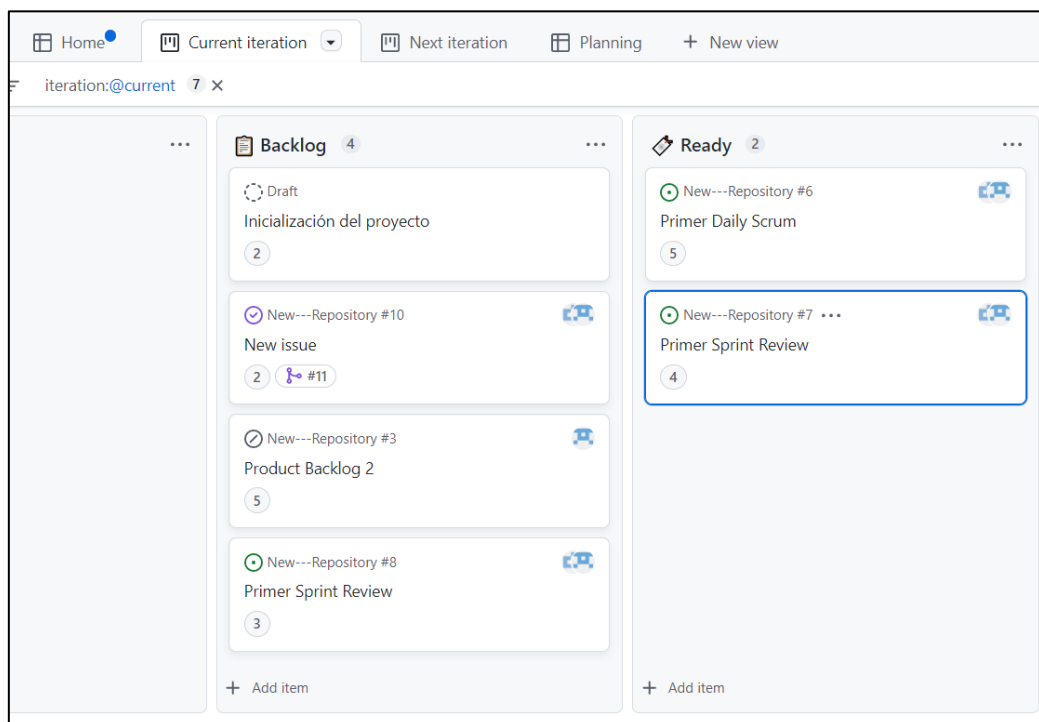
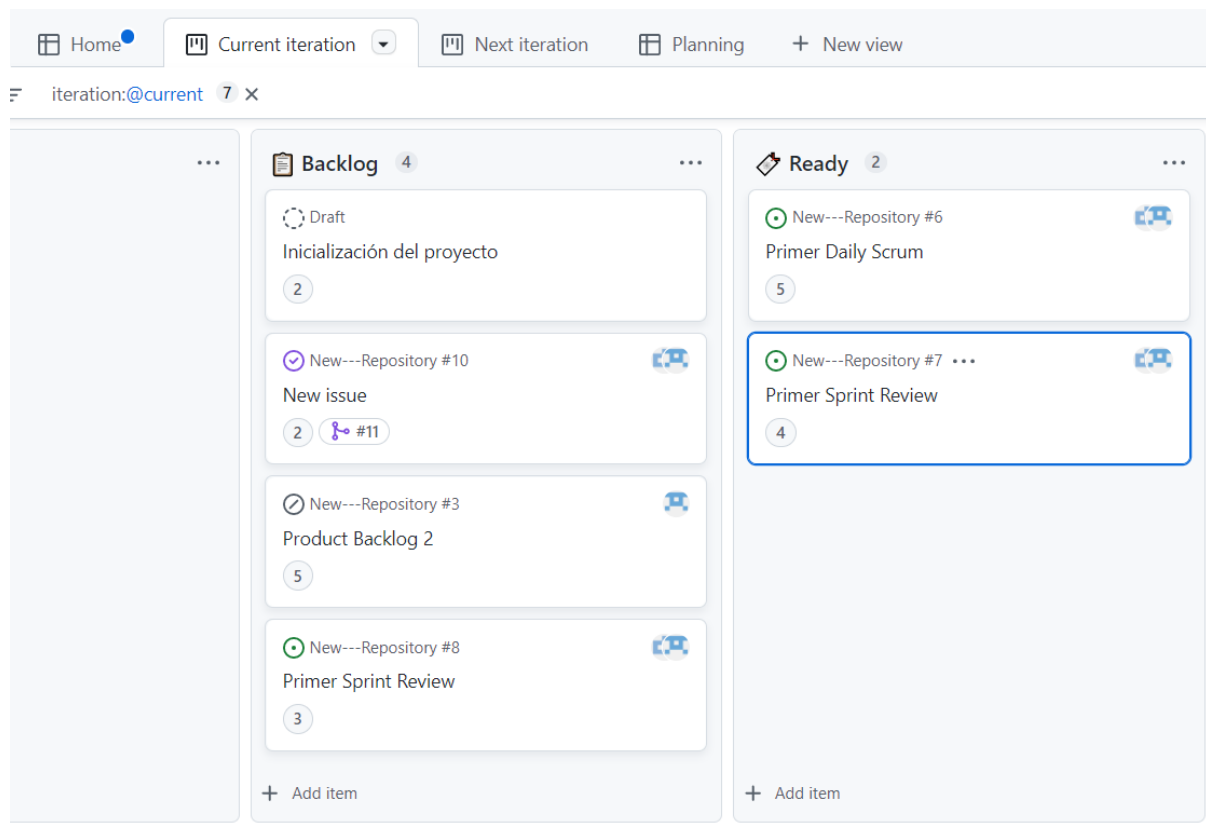


Figura 55

Asignar sprint review a la iteración (Sprint)



Cerrar un Product Backlog (Issue)

Existe dos maneras para dar como finalizado una Issue, la primera es dirigirse hacia la Issue y en la parte inferior seleccionar Close Ver Figura 56. issue. Una vez cerrado el issue es necesario ir al apartado de iteraciones en el proyecto y colocar la issue en la parte de **DONE** como se observa en la Figura 57.

Figura 56
Close Issue.

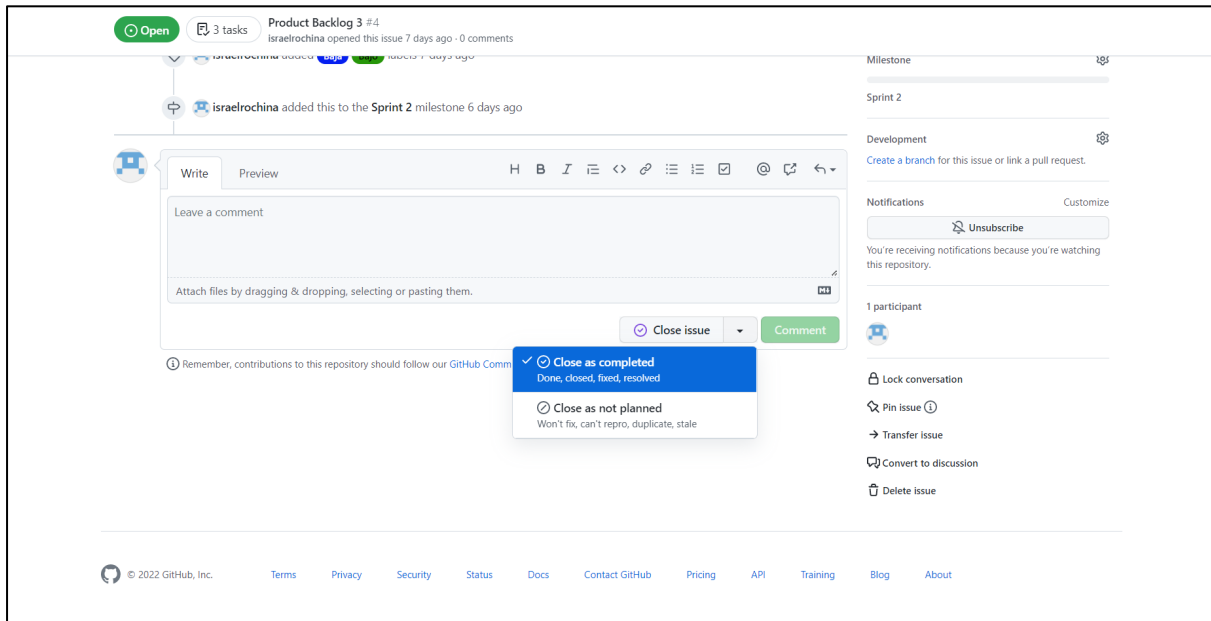
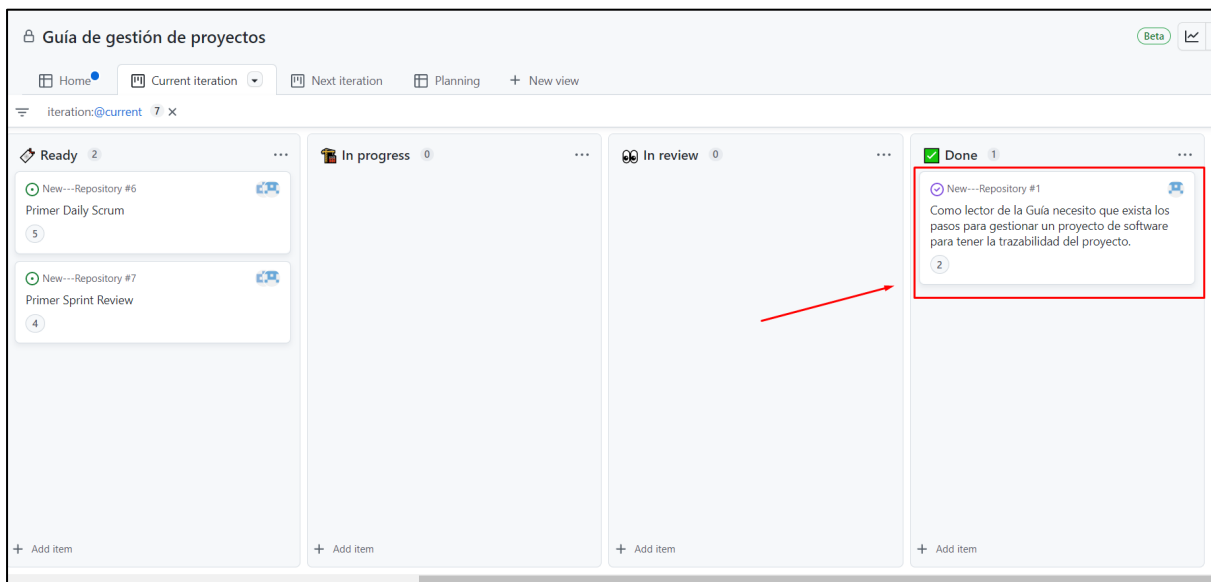


Figura 57
Finalizar issue en la iteración.



La segunda forma es realizar un commit haciendo referencia a la Issue haciendo uso de la palabra close: **git commit -m "feat: comentario - Close #1"**, el commit se debe realizar desde la rama que se creó a partir de la issue. Ver Figura 58. Luego de hacer un commit se debe subir los cambios realizados al repositorio remoto, para posteriormente realizar un Pull Request hacia la rama principal.

Figura 58

Realizar commit para cerrar una Issue.

```
Branch '10-new-issue' set up to track remote branch '10-new-issue' from 'origin'
MINGW64 ~/Desktop/New---Repository (10-new-issue)
$ git add .
MINGW64 ~/Desktop/New---Repository (10-new-issue)
$ git commit -m "Agregue nueva instrucción closes #10"
[10-new-issue df9d801] Agregue nueva instrucción closes #10
1 file changed, 2 insertions(+), 1 deletion(-)
MINGW64 ~/Desktop/New---Repository (10-new-issue)
$ git push origin 10-new-issue
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 359 bytes | 179.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/israelrochina/New---Repository.git
3366e5b..df9d801 10-new-issue -> 10-new-issue
MINGW64 ~/Desktop/New---Repository (10-new-issue)
```

Como se observa en la Figura 59 , Figura 60, Figura 61 y Figura 62 para realizar el Pull Request, ubicarse en el menú apartado **Pull request** dentro se encontrará las ramas que han dado como finalizado a una Issue. Dar clic en Compare & pull request, luego clic en Create pull request, después en merge pull request y finalmente dar clic en confirm merge.

Figura 59

Compare & pull request.



Figura 60

Create pull request.

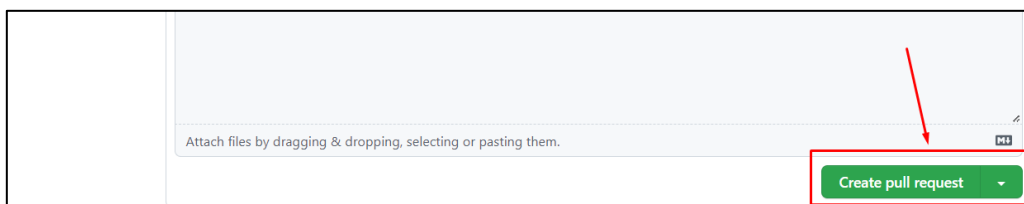


Figura 61

Merge Pull Request

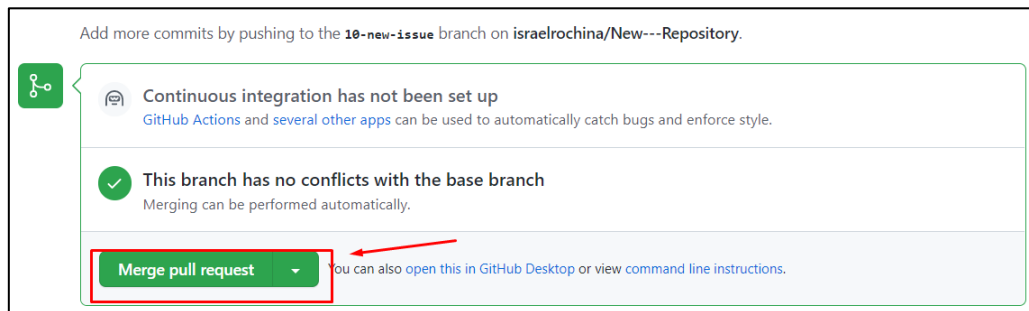
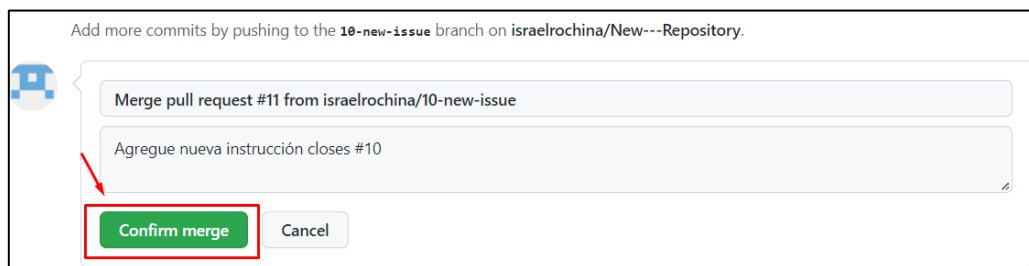


Figura 62

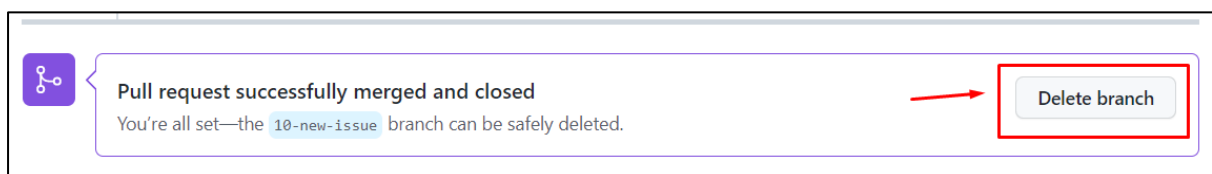
Confirm Merge.



Con el fin de reducir ramas del repositorio, una vez realizado el pull request eliminar la rama que se creó a partir de la issue. Ver Figura 63.

Figura 63

Eliminar una rama de Issue.



Realizar el Sprint Retrospective

Después de terminar el Sprint, los Developers deben dar a conocer cuáles fueron las dificultades al desarrollar el Sprint asignado. Posterior el Scrum Master planifica un incremento de calidad.

Crear un Milestone con el nombre Sprint Retrospective con el fin de organizar todo los Sprint retrospective, dentro del Sprint retrospective estarán todas las reuniones de Sprint Retrospective que se realicen hasta el fin del proyecto. Ver Figura 64. Cada iteración debe tener un Sprint retrospective como se observa en la Figura 65.

Figura 64

Creación Sprint Retrospective.

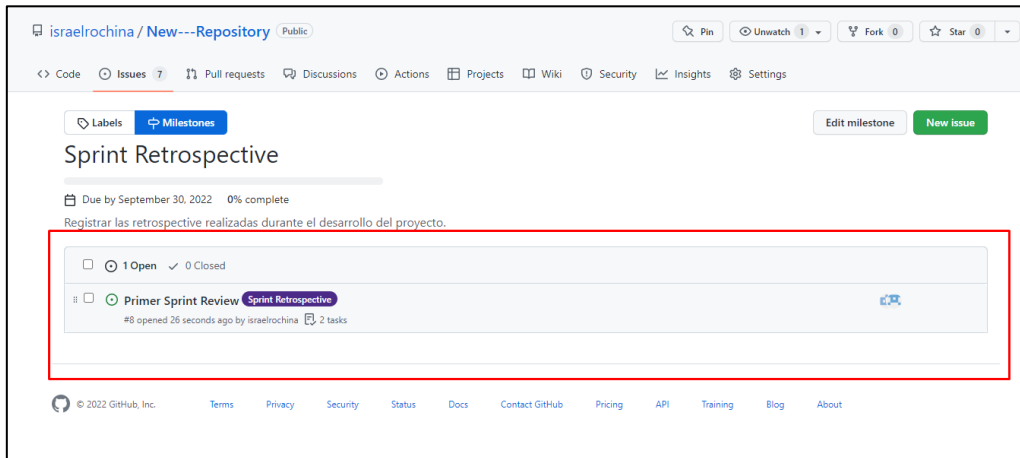
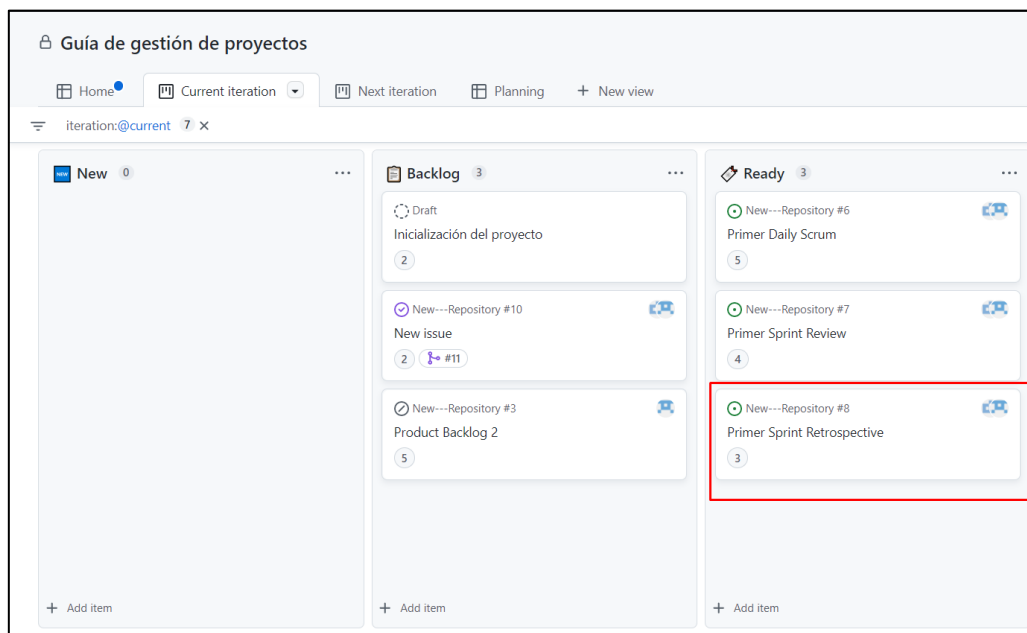


Figura 65

Asignación sprint retrospective a iteración.



3.5.3 Fase 3. Post-Juego

Increments

Es necesario guardar los incrementos que se obtuvo después de terminar un Sprint. Para gestionar los Incrementos crear un Tag, también un Releases. Los tags se crearán en N incrementos.

Crear Tag

Para crear un tag es necesario hacer por la línea de comandos desde un repositorio local que esté conectado con el repositorio remoto de GitHub. Ver Figura 66. Luego subir el tag creado al repositorio remoto. Ver Figura 67. El tag creado debe mostrar en el repositorio como en la Figura 68.

Figura 66

Crear Tag.

```
$ git tag V.01.02 -m "Finalizado la primera asignación de Sprint"
```

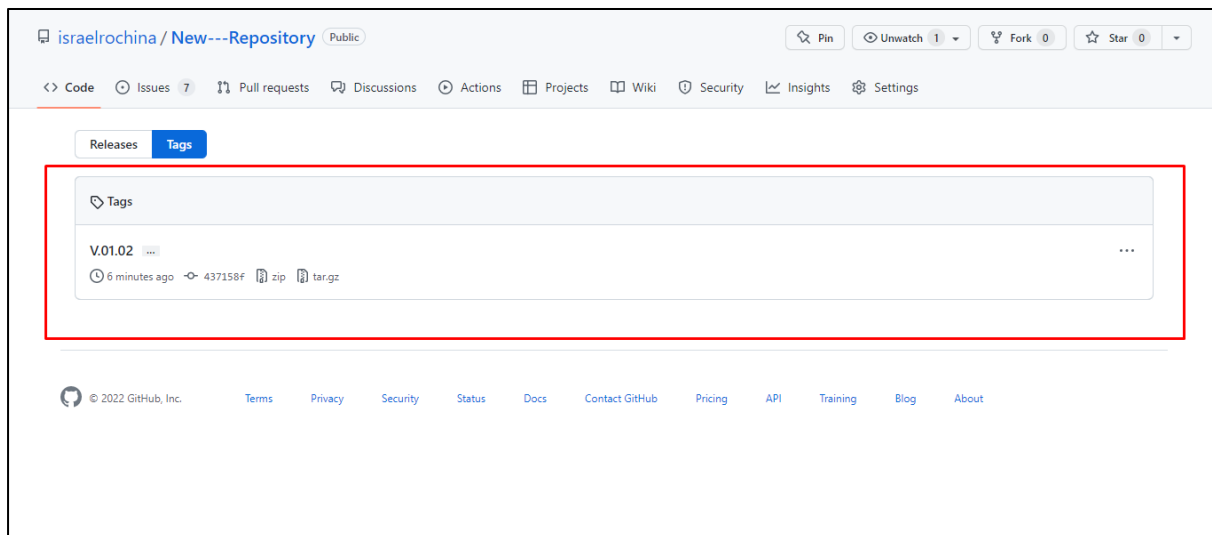
Figura 67

Subir Tag al repositorio Remoto.

```
$ git push --tags
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 197 bytes | 197.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/israelrochina/New---Repository.git
 * [new tag]          V.01.02 -> V.01.02
```

Figura 68

Tag Creado y subido al repositorio GitHub.

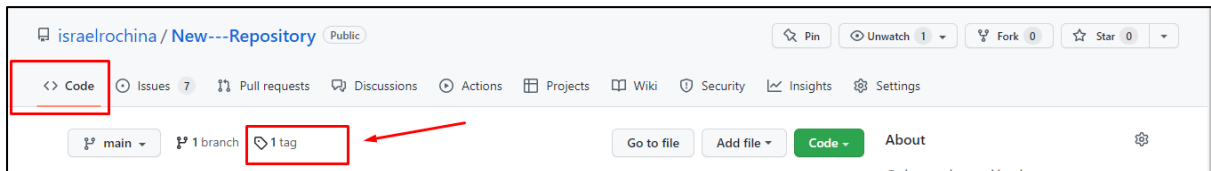


Crear Releases

Cuando el proyecto tenga un porcentaje considerable de cumplimiento de los requisitos del cliente, o cuando se termine los Sprint crear el releases. Para crear un releases diríjase al menú en el apartado de **code** de clic tag. Ver Figura 69.

Figura 69

Acceder a crear Releases.

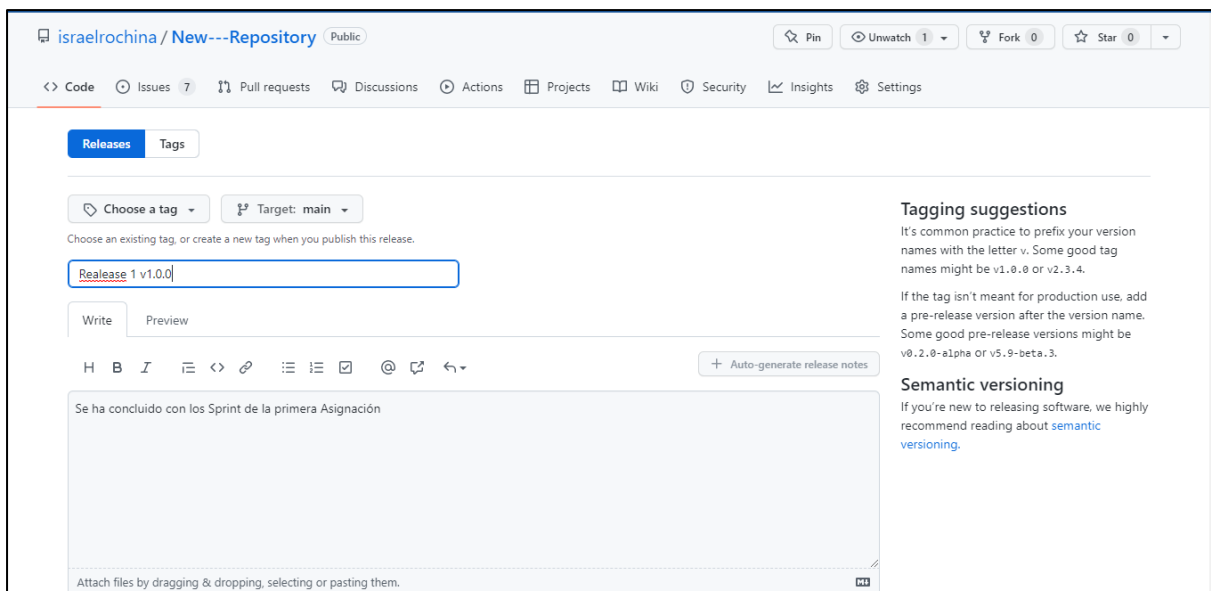


Una vez dentro Tags, dar clic en Releases y seleccionar crear New Releases. Para crear ingresar la siguiente información. Ver *Figura 70*:

- **Title:** Ingresar el nombre del Release
- **Descripción:** Ingresar el detalle de que se trata el releases
- **Seleccionar el Tag:** Seleccionar el tag

Figura 70

Creación de Releases



CAPÍTULO IV

4. Validación de resultados

Se realizó la prueba de concepto para saber el porcentaje de eficiencia que tiene la metodología realizada con GitHub y el marco de trabajo scrum. Para saber el porcentaje de eficiencia fue evaluado con la ISO/IEC 25023. La prueba de concepto consta en la elaboración de un catálogo de vehículo en el que se muestre la información necesaria para la venta de vehículo, para esto se crearan dos repositorios por un lado la parte de Backend y el otro de Frontend.

4.1. Desarrollo de la prueba de concepto

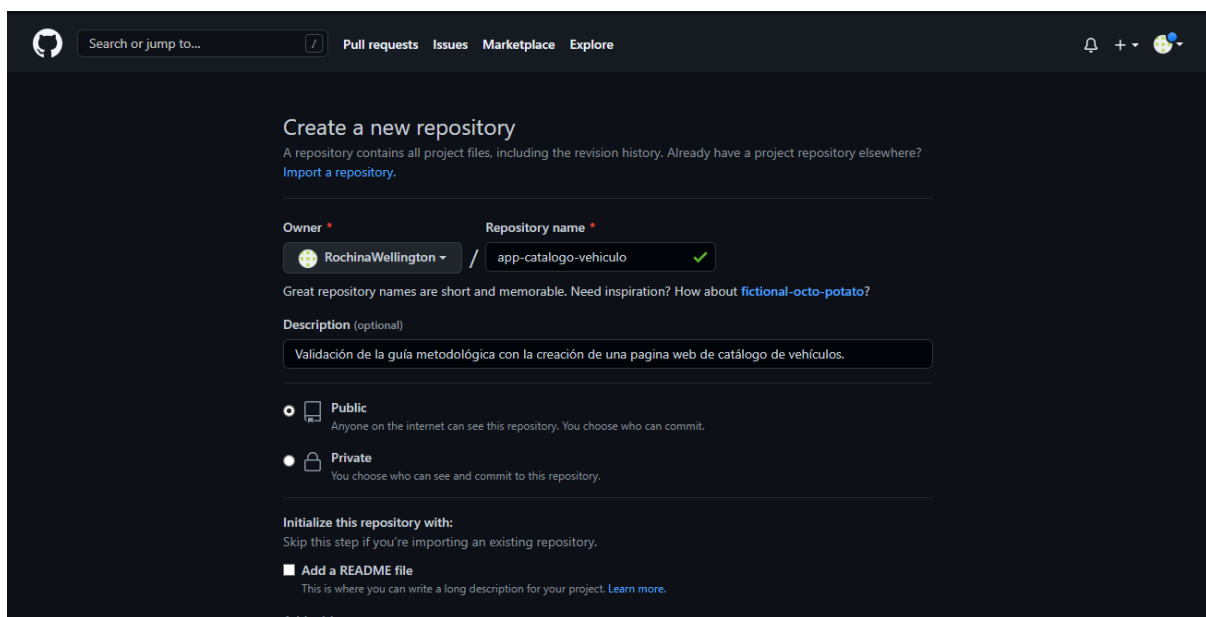
4.1.1. Fase 1. Pre-Juego

Sprint 0

Como se observa en la *Figura 71* se creó el proyecto en la herramienta GitHub el nombre asignado fue **app-catalogo-vehiculo**. El proyecto fue creado en modo Público para aprovechar todos los beneficios que ofrece la herramienta GitHub. El proyecto se integró desde la maquina local.

Figura 71

Inicialización del repositorio.



Inicializar el repositorio creado.

Para la inicialización del repositorio se realizó mediante comandos ejecutados desde la maquina local con la ayuda de **Git Bash** (Herramienta que permite ejecutar comandos de git, ayuda a interactuar con GitHub). Los comandos ejecutados fueron proporcionados por la propia herramienta GitHub ver Figura 72.

Figura 72

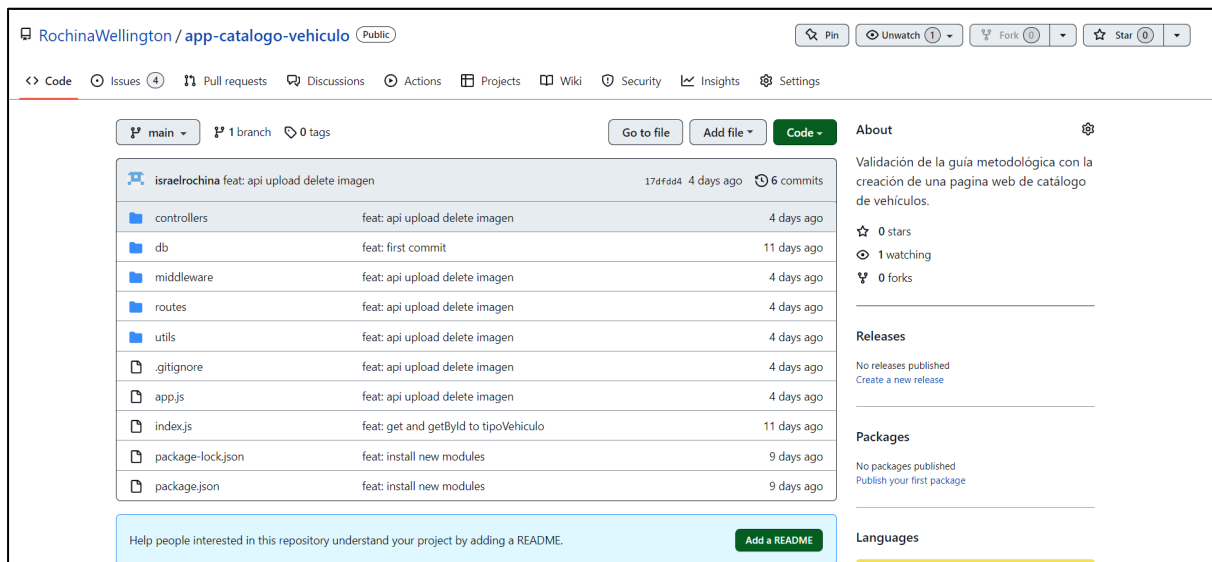
Comandos para inicializar el repositorio.

```
...or push an existing repository from the command line
git remote add origin https://github.com/RochinaWellington/app-catalogo-vehiculo.git
git branch -M main
git push -u origin main
```

Al inicializar el proyecto se agregó todos los archivos de la plantilla inicial que se encontraba en la maquina local. Estos archivos se subieron a la rama **main** que se crea por defecto, en algunas ocasiones se puede crear con el nombre **master** es dependiente de la configuración de Git. Ver Figura 73.

Figura 73

Archivos cargados desde la maquina local.



Agregar colaboradores

Los roles definidos para el desarrollo de la aplicación fueron, Product Owner, Scrum Master y Development Team.

Tabla 12

Roles dentro del proyecto.

Rol	Nombre
Product Owner	MSc. Antonio Quiña
Scrum Master	Wellington Rochina
Development Team	Wellington Rochina

Se agrego a los miembros del equipo del trabajo al repositorio de GitHub, para lo cual se envi  una solicitud de invitaci n hacia el repositorio, la solicitud es enviada por mail.

Creaci n del Product Backlog

Se creo los Product Backlog del proyecto de la prueba de concepto, estos se obtuvieron de los requerimientos iniciales que se han planteado para el proyecto. La creaci n de los Product Backlogs se realiz  en el apartado de las Issues de la herramienta de GitHub. Ver Figura 74.

Antes de crear el Product Backlog se cre  las etiquetas para identificar el nivel de dificultad que tiene los Product Backlog. **Alto, Medio, Bajo**. Tambi n se crearon etiquetas para identificar el nivel de prioridad que tiene el Product Backlog. **Alta Media Baja**. Ver Figura 75.

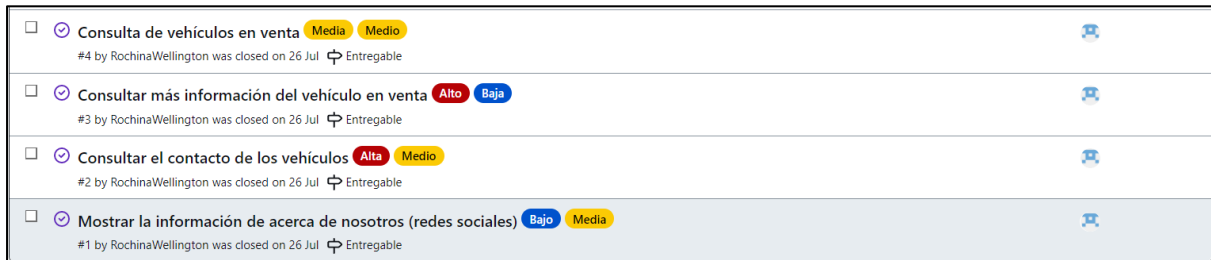
Figura 74

Etiquetas de nivel dificultad y Prioridad prueba de concepto.

Alta	Nivel de prioridad	⊙ 1	Convert issues Edit Delete
Alto	Nivel de dificultad	⊙ 1	Convert issues Edit Delete
Baja	Nivel de prioridad	⊙ 1	Convert issues Edit Delete
Bajo	Nivel de dificultad	⊙ 1	Convert issues Edit Delete
Media	Nivel de prioridad	⊙ 2	Convert issues Edit Delete
Medio	Nivel de dificultad	⊙ 2	Convert issues Edit Delete

Figura 75

Product Backlog de la prueba de concepto.



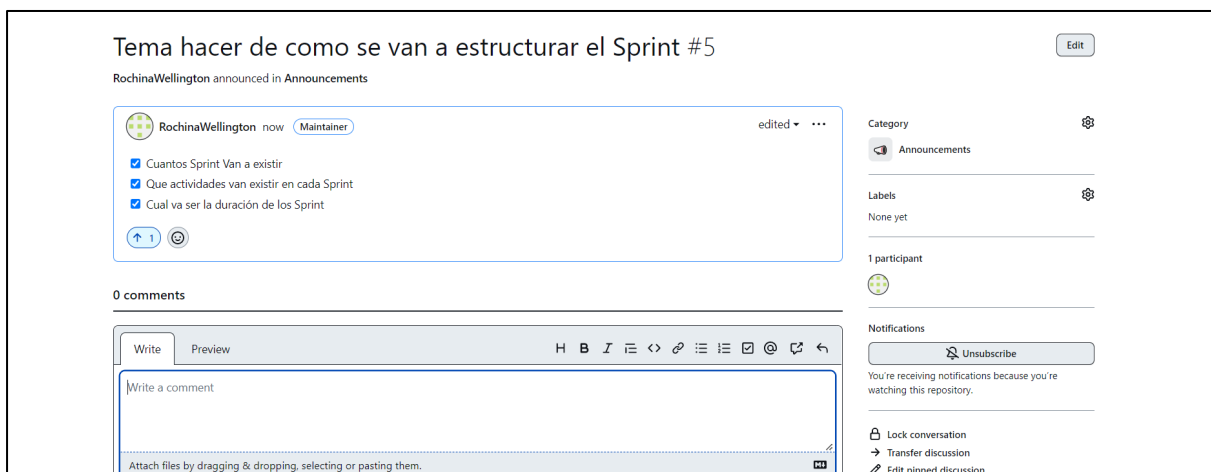
4.1.2 Fase 2. Juego

Creación de Sprint Planning

Se creó un Sprint Planning, en el que se definió el número de sprint, tiempo de ejecución y las actividades que estarían dentro de cada Sprint. Ver Figura 76.

Figura 76

Discusión de la estructura de los Sprint.



Creación de Milestone para organizar las issues

Se creó varios milestones para organizar las diferentes issues. Como se observa en la Figura 77 los milestones ayudaron a identificar de manera más fácil los objetivos que tenían los issues, también fueron de gran ayuda para organizar las tareas y dividir en secciones.

Figura 77

Creación de milestones prueba de concepto.

3 Open ✓ 4 Closed		Sort ▾
Entregable Closed 13 seconds ago ⌚ Last updated less than a minute ago Realizar un entregable de la aplicación	100% complete 0 open 4 closed Edit Reopen Delete	
Grupo Frontend Closed 15 seconds ago ⌚ Last updated less than a minute ago Generar un aplicativo Frontend consumiendo el servicio realizado la parte de backend	100% complete 0 open 8 closed Edit Reopen Delete	
Grupo Backend Closed 18 seconds ago ⌚ Last updated less than a minute ago Generar el aplicativo backend	100% complete 0 open 8 closed Edit Reopen Delete	
generación de BD Closed 27 seconds ago ⌚ Last updated less than a minute ago Generación de la base de datos. Tener un Script para un respaldo.	100% complete 0 open 4 closed Edit Reopen Delete	

Creación y gestión de Sprint

La lista de los Sprint backlog se obtuvieron de la discusión del Sprint Planning. Como resultado de la discusión se decidió crear las 4 iteraciones (Sprint). Sprint de Base de datos, Sprint de Backend, Sprint del Frontend y Sprint de entregable de la aplicación. Ver Figura 78.

Figura 78

Sprint Backlog de la prueba de concepto.

Home ▾ Current iteration Next iteration Planning Iteration 1 + New view										
Title	Assignees	Status	Estimate	Linked pull requests	Labels	Iteration				
1 <input checked="" type="checkbox"/> Realizar el entregable de la aplicación #25	israelrochina	Done	1		Alto Alto	Iteration 3				
2 <input checked="" type="checkbox"/> Mostrar la información de acerca de nosotros (redes sociales) #1	israelrochina	Done	2		Bajo Medio	Iteration 4				
3 <input checked="" type="checkbox"/> Consultar el contacto de los vehículos #2	israelrochina	Done	2		Alto Medio	Iteration 4				
4 <input checked="" type="checkbox"/> Consultar más información del vehículo en venta #3	israelrochina	Done	2		Alto Bajo	Iteration 4				
5 <input checked="" type="checkbox"/> Consulta de vehículos en venta #4	israelrochina	Done	2		Medio Medio	Iteration 4				
6 <input checked="" type="checkbox"/> Realizar el consumo del api de usuario #24	israelrochina	Done	3		Medio Medio	Iteration 3				
7 <input checked="" type="checkbox"/> Realizar el consumo del api de tipo de vehículo #23	israelrochina	Done	3		Bajo Medio	Iteration 3				
8 <input checked="" type="checkbox"/> Realizar la conexión al servicio de backend #18	israelrochina	Done	1		Bajo Medio	Iteration 3				
9 <input checked="" type="checkbox"/> Realizar el consumo de modelo de vehículo #22	israelrochina	Done	3		Medio Medio	Iteration 3				
10 <input checked="" type="checkbox"/> Realizar el consumo de la api de modelo #21	israelrochina	Done	1		Alto Medio	Iteration 3				
11 <input checked="" type="checkbox"/> Realizar el consumo del api de vehículos #20	israelrochina	Done	2		Alto Alto	Iteration 3				
12 <input checked="" type="checkbox"/> realizar el consumo de la api de Usuarios #19	israelrochina	Done	2		Alto Medio	Iteration 3				
13 <input checked="" type="checkbox"/> Configuración de backend con servidor de express #10	israelrochina	Done	2		Alto Bajo	Iteration 2				

Como se observa en la Figura 79 los sprint están divididos 4 semanas, cada sprint tiene la duración de una semana. La fecha de inicio fue del 29 de junio hasta la fecha fin de 26 de julio de 2022.

Figura 79

Iteraciones creadas para la prueba de concepto.

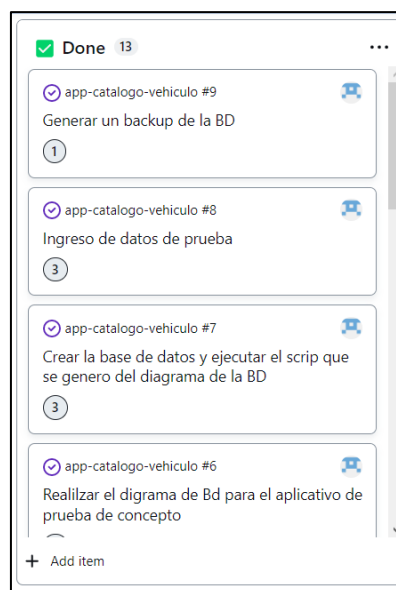
🕒 1 Active 📅 3 Completed	
Completed	Iteration 1 1 week Jun 29 - Jul 05
Completed	Iteration 2 1 week Jul 06 - Jul 12
Completed	Iteration 3 1 week Jul 13 - Jul 19

Sprint 1

Como se observa en la Figura 80 dentro del sprint se propuso diseñar la BD, posterior a la validación y aprobación se estableció crear la BD en PostgreSQL y cargar el diseño hacia la BD creado. También se planteó cargar información de prueba a todas las Tablas de la BD.

Figura 80

Sprint 1 de la prueba de concepto.

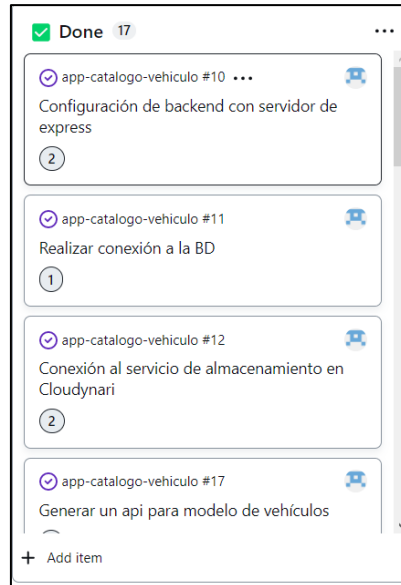


Sprint 2

Como se observa en la Figura 81 en el sprint 2 se planteó hacer la conexión de base de datos con la parte de Backend, realizar las APIS de vehículo, modelo, tipo de vehículo, contacto o usuario finalmente se planteó subir a un repositorio de Heroku.

Figura 81

Sprint 2 de la prueba de concepto.

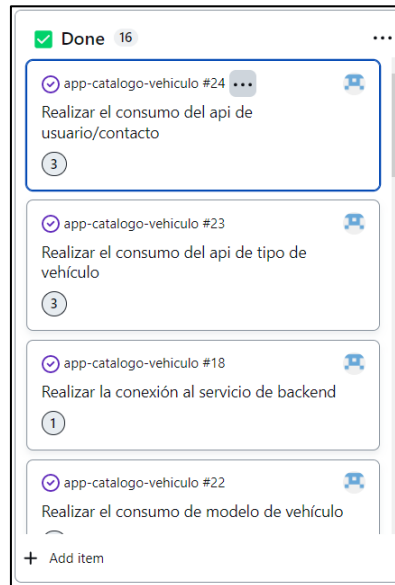


Sprint 3

Como se observa en la Figura 82, para el sprint 3 se planteó realizar toda la parte de Frontend: conexión de Backend al Frontend, consumo del api de Backend, realizar los diseños del encabezado y pie, realizar una vista para ver más información del vehículo y contacto.

Figura 82

Sprint 3 de la prueba de concepto.

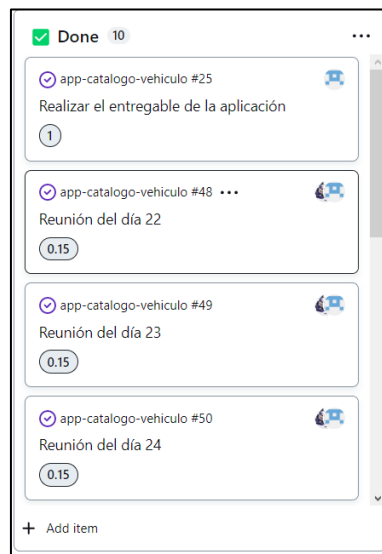


Sprint 4

Como se observa en la Figura 83 en el sprint 4 se realizó el entregable para este caso el entregable fue la finalización del proyecto. Dentro del entregable se incluyeron las reuniones que se realizaban cada finalización del sprint con el fin de revisar el cumplimiento de este.

Figura 83

Sprint 4 de la prueba de concepto.



Ejecución de la iteración

a) Asignación de tareas

Como se observa en la Figura 84, se realizó la asignación de tareas en la columna de Assignees. Se puede asignar a más de una persona en una tarea.

Figura 84

Asignación de tareas prueba de concepto

	Title	Assignees
1	Realizar el entregable de la aplicación #25	israelrochina...
2	Mostrar la información de acerca de nosotros (redes sociales) #1	Search people
3	Consultar el contacto de los vehículos #2	RochinaWelling
4	Consultar más información del vehículo en venta #3	israelrochina
5	Consulta de vehículos en venta #4	israelrochina
6	Realizar el consumo del api de usuario/contacto #24	israelrochina
7	Realizar el consumo del api de tipo de vehículo #23	israelrochina
8	Realizar la conexión al servicio de backend #18	israelrochina
9	Realizar el consumo de modelo de vehículo #22	israelrochina
10	Realizar el consumo de la api de modelo #21	israelrochina
11	Realizar el consumo del api de vehículos #20	israelrochina
12	realizar el consumo de la api de Usuarios #19	israelrochina
13	Configuración de backend con servidor de express #10	israelrochina

b) Ejecución de la issues por parte de los developers

- **Sprint 1:** El desarrollo del Sprint se ha realizado en un tiempo de 11 horas. la fecha de desarrollo fue en (29 junio del 2022 – 5 de julio del 2022). Ver Figura 85.

Figura 85

Realización del sprint 1 prueba de concepto

	Title	Assignees	Status	Estimate	Linked pull requests	Labels	Iteration
1	Generar un backup de la BD #9	israelrochina	Done	1		Alta	Iteration 1
2	Ingreso de datos de prueba #8	israelrochina	Done	3		Medio	Iteration 1
3	Crear la base de datos y ejecutar el scrip que se genero del diag #7	israelrochina	Done	3		Alta Alto	Iteration 1
4	Realizar el digrama de Bd para el aplicativo de prueba de conce #6	israelrochina	Done	4		Alta	Iteration 1
5	Reunión del día 2 #28	israelrochina...	Done	0.15		Alta	Iteration 1
6	Reunión del día 3 #29	israelrochina...	Done	0.15		Alto	Iteration 1
7	Reunión del día 4 #30	israelrochina...	Done	0.15		Alto	Iteration 1
8	Reunión del día 6 #32	israelrochina...	Done	0.15		Alto	Iteration 1
9	Reunión del día 7 #33	israelrochina...	Done	0.15		Alto	Iteration 1
10	Reunión del día 5 #31	israelrochina...	Done	0.15		Alto	Iteration 1
11	Reunión del día 1 #27	israelrochina...	Done	0.15		Alto	Iteration 1
12	Sprint review de iteración 1 #55	israelrochina...	Done	2		Alto	Iteration 1
13	Sprint retrospective de iteración 1 #59	israelrochina...	Done	2		Alto	Iteration 1

- **Sprint 2:** El desarrollo del Sprint se ha realizado en un tiempo de 15 horas. la fecha de desarrollo fue en **(6 de julio de 2022 – 12 de julio de 2022)**. Ver Figura 86.

Figura 86

Realización del sprint 2 prueba de concepto.

Title	Assignees	Status	Estimate	Linked pull requests	Labels	Iteration
14 Configuración de backend con servidor de express #10	israelrochina	Done	2		Alta Bajo	Iteration 2
15 Realizar conexión a la BD #11	israelrochina	Done	1		Alta Alto	Iteration 2
16 Conexión al servicio de almacenamiento en Cloudynari #12	israelrochina	Done	2		Alta Alto	Iteration 2
17 Generar un api para modelo de vehículos #17	israelrochina	Done	1		Alta	Iteration 2
18 Generar api para Marca #16	israelrochina	Done	1		Media Me	Iteration 2
19 Realizar api de get para tipo de vehículo #15	israelrochina	Done	2			Iteration 2
20 Realizar Crud de Venta de Vehículos #14	israelrochina	Done	3		Alta	Iteration 2
21 Realizar Crud para Usuarios #13	israelrochina	Done	3		Alta	Iteration 2
22 Reunión del día 8 #34	israelrochina	Done	0.15		Alto	Iteration 2
23 Reunión del día 9 #35	israelrochina	Done	0.15		Alto	Iteration 2
24 Reunión del día 10 #36	israelrochina	Done	0.15		Alto	Iteration 2
25 Reunión del día 11 #37	israelrochina	Done	0.15		Alto	Iteration 2
26 Reunión del día 13 #38	israelrochina	Done	0.15		Alto	Iteration 2

- **Sprint 3:** El desarrollo del Sprint se ha realizado en un tiempo de 16 horas. la fecha de desarrollo fue en **(13 de julio de 2022 – 19 de julio de 2022)**. Ver Figura 87.

Figura 87

Realización del sprint 3 prueba de concepto.

Title	Assignees	Status	Estimate	Linked pull requests	Labels	Iteration
31 Mostrar la información de acerca de nosotros (redes sociales) #1	israelrochina	Done	2		Bajo Medi	Iteration 3
32 Consultar el contacto de los vehículos #2	israelrochina	Done	2		Alta Medic	Iteration 3
33 Consultar más información del vehículo en venta #3	israelrochina	Done	2		Alto Bajo	Iteration 3
34 Consulta de vehículos en venta #4	israelrochina	Done	2		Media Me	Iteration 3
35 Realizar el consumo del api de usuario/contacto #24	israelrochina	Done	3		Media Me	Iteration 3
36 Realizar el consumo del api de tipo de vehículo #23	israelrochina	Done	3		Bajo Medi	Iteration 3
37 Realizar la conexión al servicio de backend #18	israelrochina	Done	1		Bajo Medi	Iteration 3
38 Realizar el consumo de modelo de vehículo #22	israelrochina	Done	3		Media Me	Iteration 3
39 Realizar el consumo de la api de modelo #21	israelrochina	Done	1		Alto Medic	Iteration 3
40 Realizar el consumo del api de vehículos #20	israelrochina	Done	2		Alta Alto	Iteration 3
41 realizar el consumo de la api de Usuarios #19	israelrochina	Done	2		Alta Medic	Iteration 3
42 Reunión del día 15 #41	israelrochina	Done	0.15		Alto	Iteration 3
43 Reunión del día 16 #42	israelrochina	Done	0.15		Alto	Iteration 3

- **Sprint 4:** El desarrollo del Sprint se ha realizado en un tiempo de 13:45 horas. la fecha de desarrollo fue en **(13 de julio de 2022 – 19 de julio de 2022)**. Ver Figura 88.

Figura 88

Realización del sprint 4 prueba de concepto.

Title	Assignees	Status	Estimate	Linked pull requests	Labels	Iteration
1 Sprint retrospective de iteración 4 #62	israelrochin...	Done	2		Alto	Iteration 4
2 Sprint review de iteración 4 #58	israelrochin...	Done	2		Alto	Iteration 4
3 Reunión del día 28 #54	israelrochin...	Done	0.15		Alto	Iteration 4
4 Reunión del día 27 #53	israelrochin...	Done	0.15		Alto	Iteration 4
5 Reunión del día 26 #52	israelrochin...	Done	0.15		Alto	Iteration 4
6 Reunión del día 25 #51	israelrochin...	Done	0.15		Alto	Iteration 4
7 Reunión del día 24 #50	israelrochin...	Done	0.15		Alto	Iteration 4
8 Reunión del día 23 #49	israelrochin...	Done	0.15		Alto	Iteration 4
9 Reunión del día 22 #48	israelrochin...	Done	0.15		Alto	Iteration 4
10 Realizar el entregable de la aplicación #25	israelrochin...	Done	1		Alto Alto	Iteration 4

c) Planificar Daily Scrum

Como se observa en la Figura 89 se creó un milestone para gestionar las reuniones diarias, por cada reunión se creó su respectivo issue. Ver Figura 90, Figura 91, Figura 92 y Figura 93.

Figura 89

Daily scrum prueba de concepto.

Labels
Milestones

Daily Scrum

No due date 100% complete

Reuniones realizados durante el desarrollo de la prueba de concepto.
Los miembros del equipo de desarrollo solventan dudas, se planifica cuales son las proximas actividades.

Sprint 1

Figura 90

Daily scrum sprint 1 prueba de concepto

Title	Assignees	Status	Estimate	Labels	Iteration	Milestone
25 Reunión del día 1	israelrochin...	Done	0.15	Alto	Iteration 1	Daily Scrum
26 Reunión del día 2	israelrochin...	Done	0.15	Alto	Iteration 1	Daily Scrum
27 Reunión del día 3	israelrochin...	Done	0.15	Alto	Iteration 1	Daily Scrum
28 Reunión del día 4	israelrochin...	Done	0.15	Alto	Iteration 1	Daily Scrum
29 Reunión del día 5	israelrochin...	Done	0.15	Alto	Iteration 1	Daily Scrum
30 Reunión del día 6	israelrochin...	Done	0.15	Alto	Iteration 1	Daily Scrum
31 Reunión del día 7	israelrochin...	Done	0.15	Alto	Iteration 1	Daily Scrum

Sprint 2

Figura 91

Daily scrum sprint 2 prueba de concepto.

milestone:"Daily Scrum" iteration:"Iteration 2" 7 X								
Title	↑ ...	Assignees ...	Status ...	Estimate ...	Labels ...	Iteration ...	Milestone ...	
26	✓	Reunión del día 10	israelrochin...	Done	0.15	Alto	Iteration 2	Daily Scrum
27	✓	Reunión del día 11	israelrochin...	Done	0.15	Alto	Iteration 2	Daily Scrum
28	✓	Reunión del día 12	israelrochin...	Done	0.15	Alto	Iteration 2	Daily Scrum
29	✓	Reunión del día 13	israelrochin...	Done	0.15	Alto	Iteration 2	Daily Scrum
30	✓	Reunión del día 14	israelrochin...	Done	0.15	Alto	Iteration 2	Daily Scrum
37	✓	Reunión del día 8	israelrochin...	Done	0.15	Alto	Iteration 2	Daily Scrum
38	✓	Reunión del día 9	israelrochin...	Done	0.15	Alto	Iteration 2	Daily Scrum

Sprint 3

Figura 92

Daily scrum sprint 3 prueba de concepto.

milestone:"Daily Scrum" iteration:"Iteration 3" 7 X								
Title	↑ ...	Assignees ...	Status ...	Estimate ...	Labels ...	Iteration ...	Milestone ...	
31	✓	Reunión del día 15	israelrochin...	Done	0.15	Alto	Iteration 3	Daily Scrum
32	✓	Reunión del día 16	israelrochin...	Done	0.15	Alto	Iteration 3	Daily Scrum
33	✓	Reunión del día 17	israelrochin...	Done	0.15	Alto	Iteration 3	Daily Scrum
34	✓	Reunión del día 18	israelrochin...	Done	0.15	Alto	Iteration 3	Daily Scrum
35	✓	Reunión del día 19	israelrochin...	Done	0.15	Alto	Iteration 3	Daily Scrum
37	✓	Reunión del día 20	israelrochin...	Done	0.15	Alto	Iteration 3	Daily Scrum
38	✓	Reunión del día 21	israelrochin...	Done	0.15	Alto	Iteration 3	Daily Scrum

Sprint 4

Figura 93

Daily scrum sprint 4 prueba de concepto.

milestone:"Daily Scrum" iteration:"Iteration 4" 7 X								
Title	↑ ...	Assignees ...	Status ...	Estimate ...	Labels ...	Iteration ...	Milestone ...	
39	✓	Reunión del día 22	israelrochin...	Done	0.15	Alto	Iteration 4	Daily Scrum
40	✓	Reunión del día 23	israelrochin...	Done	0.15	Alto	Iteration 4	Daily Scrum
41	✓	Reunión del día 24	israelrochin...	Done	0.15	Alto	Iteration 4	Daily Scrum
42	✓	Reunión del día 25	israelrochin...	Done	0.15	Alto	Iteration 4	Daily Scrum
43	✓	Reunión del día 26	israelrochin...	Done	0.15	Alto	Iteration 4	Daily Scrum
44	✓	Reunión del día 27	israelrochin...	Done	0.15	Alto	Iteration 4	Daily Scrum
45	✓	Reunión del día 28	israelrochin...	Done	0.15	Alto	Iteration 4	Daily Scrum

d) Realizar Sprint Review

Para gestionar los Sprint Review se creó un milestone donde se agregó las issues correspondientes al Sprint Review. Ver Figura 94. Cada Sprint review fue asignado a su iteración correspondiente. Ver Figura 95.

Figura 94

Sprint review prueba de concepto

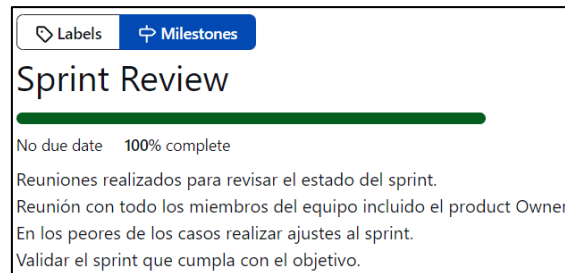


Figura 95

Sprint review asignado a las iteraciones prueba de concepto.

Title	Assignees	Status	Estimate	Labels	Iteration	Milestone
53 <input checked="" type="checkbox"/> Sprint review de iteración 1	israelrochin...	<input checked="" type="checkbox"/> Done	2	Alto	Iteration 1	Sprint Review
54 <input checked="" type="checkbox"/> Sprint review de iteración 2	israelrochin...	<input checked="" type="checkbox"/> Done	2	Alto	Iteration 2	Sprint Review
55 <input checked="" type="checkbox"/> Sprint review de iteración 3	israelrochin...	<input checked="" type="checkbox"/> Done	2	Alto	Iteration 3	Sprint Review
56 <input checked="" type="checkbox"/> Sprint review de iteración 4	israelrochin...	<input checked="" type="checkbox"/> Done	2	Alto	Iteration 4	Sprint Review

e) Realizar el Sprint Retrospective

Para gestionar los Sprint retrospective se creó un milestone con el nombre sprint retrospective como se observa en la *Figura 96*. También se creó los sprint retrospective por cada iteración. Ver *Figura 97*.

Figura 96

Sprint retrospective prueba de concepto.

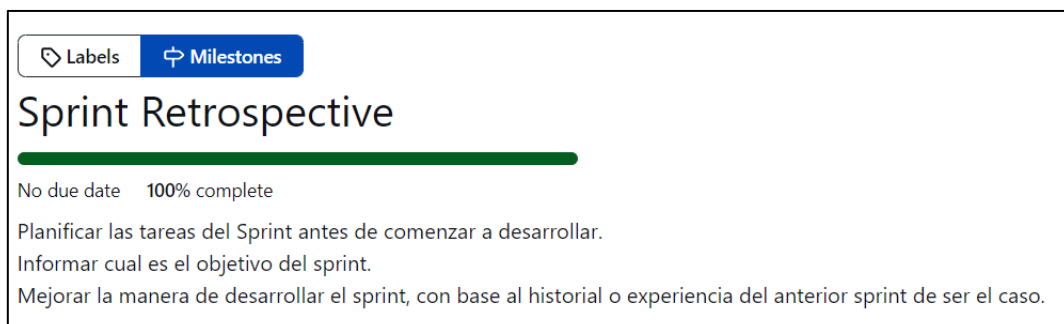


Figura 97

Sprint retrospective asignado a las iteraciones prueba de concepto.

Title	Assignees	Status	Estimate	Labels	Iteration	Milestone
53 Sprint retrospective de iteración 1	israelrochin...	Done	2	Alto	Iteration 1	Sprint Retrospective
54 Sprint retrospective de iteración 2	israelrochin...	Done	2	Alto	Iteration 2	Sprint Retrospective
55 Sprint retrospective de iteración 3	israelrochin...	Done	2	Alto	Iteration 3	Sprint Retrospective
56 Sprint retrospective de iteración 4	israelrochin...	Done	2	Alto	Iteration 4	Sprint Retrospective

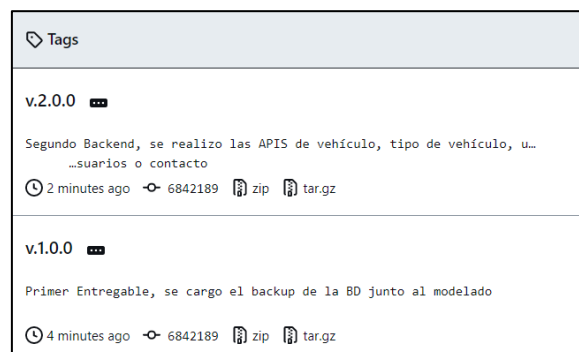
4.1.3. Fase 3. Post-Juego

Increments

Se creo varios incrementos dependiendo de los avances y cumplimiento del objetivo. Antes de crear el incremento fue necesario hacer una revisión del código que estuviese todo en orden y no exista inconveniente con otras ramas del proyecto. Como se observa en la Figura 98 para plasmar los incrementos se creó los tags.

Figura 98

Tags de la prueba de concepto.



Una vez finalizado el proyecto se creó un **Release** que fue el entregable final donde se encontraba la codificación del proyecto. El release se generó a partir de la última versión del proyecto. Ver *Figura 99*.

Figura 99

Entregable final de la prueba de concepto.

Entregable Latest

Compare ✎ 🗑️

RochinaWellington released this 1 minute ago · 0 commits to main since this release · v2.0.0 · 43ba2a5

La versión es estable y fue aprobado por los miembros del comité.
Se da por finalizado el proyecto, se realizó un nuevo acuerdo para establecer fecha de soporte a la app.

▼ Assets 2

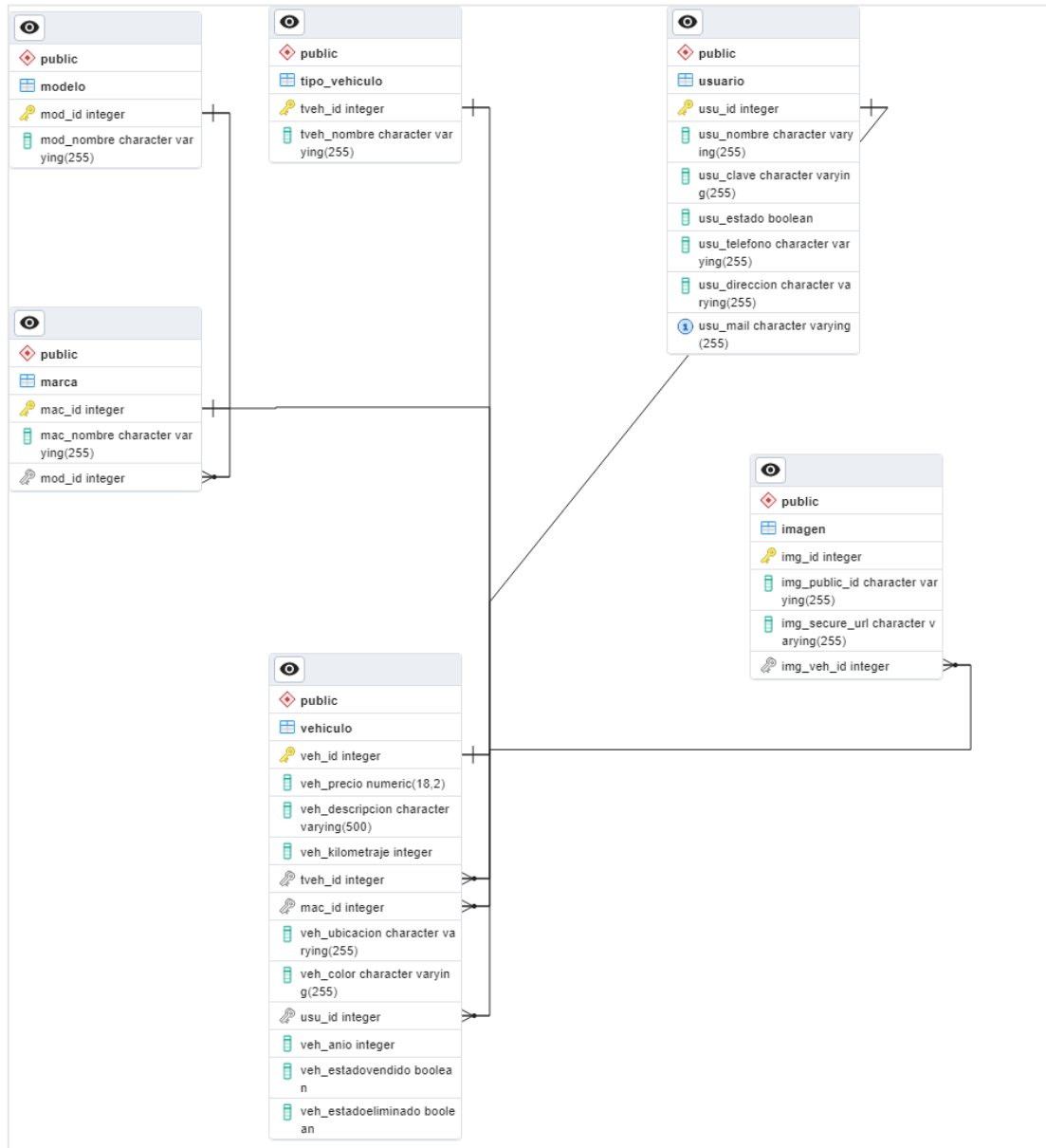
Source code (zip)	3 minutes ago
Source code (tar.gz)	3 minutes ago

1 1 You reacted

Anexo A: Diagrama de base de datos realizado en postgres de la prueba de concepto

Figura 100

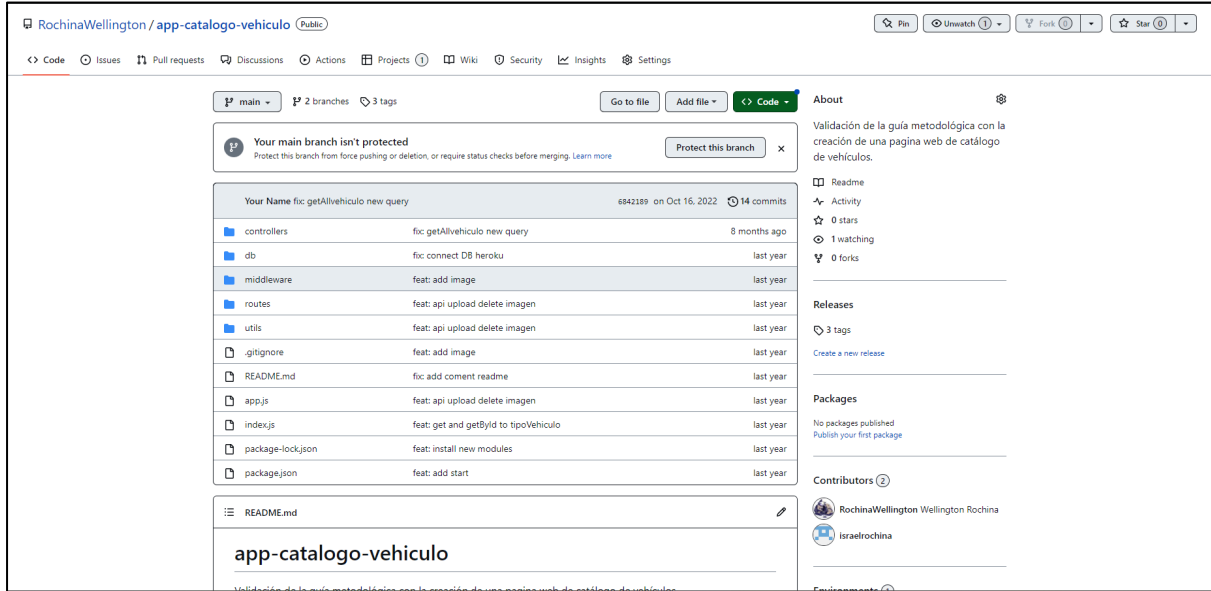
Diagrama de BD prueba de concepto.



Anexo B: Repositorio de la parte backend en GitHub.

Figura 101

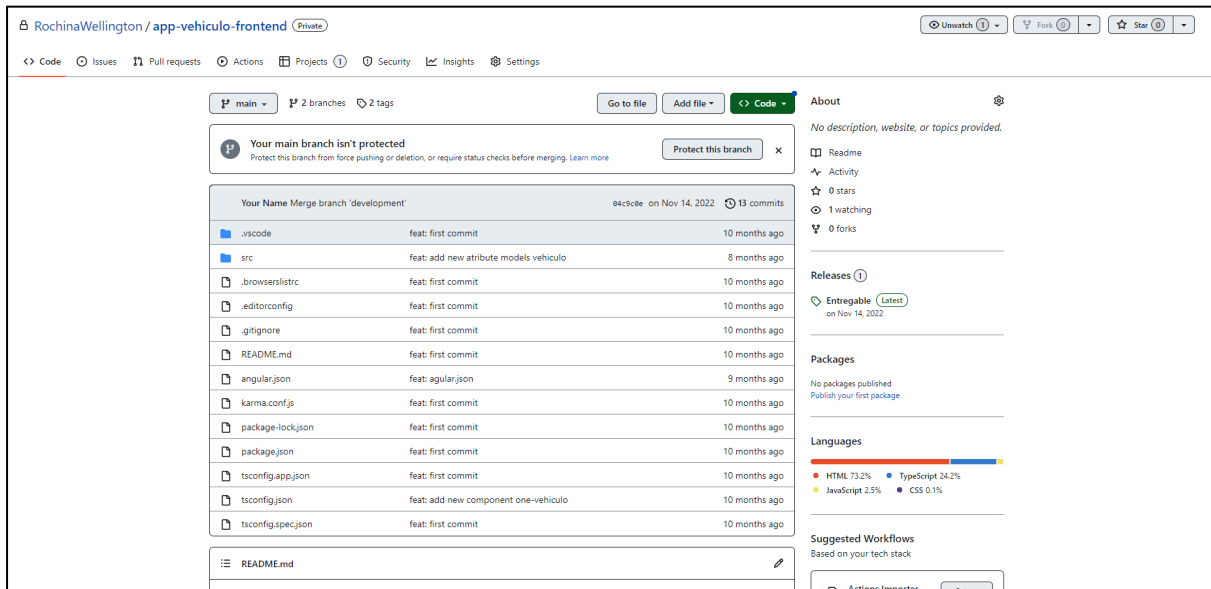
Repositorio backend prueba concepto.



Anexo C: Repositorio de la parte Frontend en GitHub

Figura 102

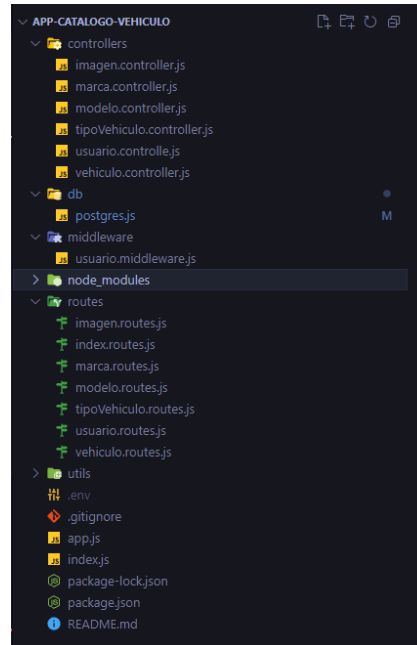
Repositorio frontend prueba concepto



Anexo D: Arquitectura de la parte backend

Figura 103

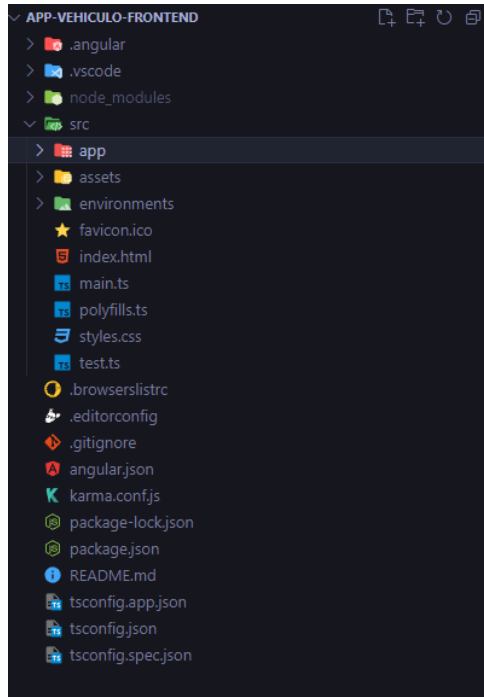
Arquitectura backend prueba de concepto.



Anexo E: Arquitectura de la parte frontend

Figura 104

Arquitectura frontend prueba de concepto



Anexo F: Gestión del proyecto.

Figura 105

Gestión de proyecto prueba de concepto.

@RochinaWellington's feature									
Home Current iteration Next iteration Planning Iteration 1 + New View									
Filter by keyword or by field									
Title	Assignees	Status	Estimate	Linked pull requests	Labels	Iteration			
1 Realizar el entregable de la aplicación #25	israelrochin...	Done	8		Alto Alto	Iteration 4			
2 Mostrar la información de acerca de nosotros (redes sociales) #1	israelrochina	Done	2		Bajo Medi	Iteration 3			
3 Consultar el contacto de los vehículos #2	israelrochina	Done	2		Alto Medi	Iteration 3			
4 Consultar más información del vehículo en venta #3	israelrochina	Done	2		Alto Bajo	Iteration 3			
5 Consulta de vehículos en venta #4	israelrochina	Done	2		Media Me	Iteration 3			
6 Realizar el consumo del api de usuario/contacto #24	israelrochina	Done	3		Media Me	Iteration 3			
7 Realizar el consumo del api de tipo de vehículo #23	israelrochina	Done	3		Bajo Medi	Iteration 3			
8 Realizar la conexión al servicio de backend #18	israelrochina	Done	1		Bajo Medi	Iteration 3			
9 Realizar el consumo de modelo de vehículo #22	israelrochina	Done	3		Media Me	Iteration 3			
10 Realizar el consumo de la api de modelo #21	israelrochina	Done	1		Alto Medi	Iteration 3			
11 Realizar el consumo del api de vehículos #20	israelrochina	Done	2		Alto Alto	Iteration 3			
12 realizar el consumo de la api de Usuarios #19	israelrochina	Done	2		Alto Medi	Iteration 3			
13 Configuración de backend con servidor de express #10	israelrochina	Done	2		Alto Bajo	Iteration 2			
14 Realizar conexión a la BD #11	israelrochina	Done	1		Alto Alto	Iteration 2			
15 Conexión al servicio de almacenamiento en Cloudynari #12	israelrochina	Done	2		Alto Alto	Iteration 2			
16 Generar un api para modelo de vehículos #17	israelrochina	Done	1		Alto	Iteration 2			
17 Generar api para Marca #16	israelrochina	Done	1		Media Me	Iteration 2			
18 Realizar api de get para tipo de vehículo #15	israelrochina	Done	2			Iteration 2			

+ You can use **Control + Space** to add an item

Anexo G: Servicios rest.

Figura 106

Servicio rest prueba de concepto.

The screenshot shows a REST client interface with a GET request to `http://localhost:3000/vehiculo`. The response is a JSON array of two vehicle objects. The first object is for a Land Rover and the second is for a Toyota. The JSON data is as follows:

```
[{"veh_id": 8, "veh_precio": "22499.74", "veh_descripcion": "Auto compacto en excelente estado: bajo kilometraje, econ\u00f3mico en combustible, ideal para la ciudad. \u00a1No te lo pierdas!", "veh_kilometraje": 47426, "tveh_id": 5, "mac_id": 9, "veh_ubicacion": "2546 Portage Place", "veh_color": "khaki", "usu_id": 4, "veh_a\u00f1o": 2012, "veh_estadovendido": false, "veh_estadoeliminado": false, "mac_nombre": "Land Rover"}, {"veh_id": 11, "veh_precio": "15796.68", "veh_descripcion": "Oportunidad imperdible: furgoneta vers\u00e1til, ideal para transporte de carga o como veh\u00edculo comercial. \u00a1Potencia y funcionalidad!\u00a1", "veh_kilometraje": 6136, "tveh_id": 2, "mac_id": 11, "veh_ubicacion": "74 Prairieview Street", "veh_color": "Tuxquoise", "usu_id": 7, "veh_a\u00f1o": 2018, "veh_estadovendido": false, "veh_estadoeliminado": false, "mac_nombre": "Toyota"}]
```

Anexo H: Pagina Web inicio catalogo veh\u00edculos.

Figura 107

P\u00e1gina web inicio cat\u00e1logo veh\u00edculos

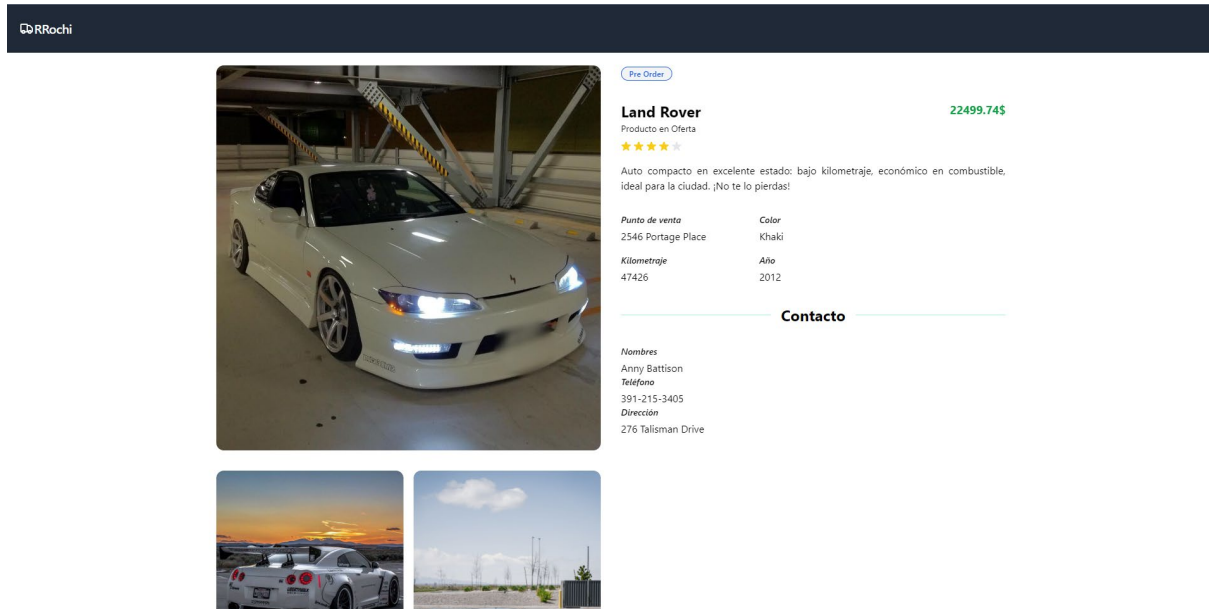
The screenshot shows a web page with a grid of car listings. Each listing includes a car image, a red banner with the word "Oferta", the car brand, price, description, year, and mileage, and a "M\u00e1s informaci\u00f3n" button. The listings are for Land Rover, Toyota, and BMW.

Brand	Price	Description	Year	Mileage
Land Rover	22499.745	Auto compacto en excelente estado: bajo kilometraje, econ\u00f3mico en combustible, ideal para la ciudad. \u00a1No te lo pierdas!	2012	47426
Toyota	15796.685	Oportunidad imperdible: furgoneta vers\u00e1til, ideal para transporte de carga o como veh\u00edculo comercial. \u00a1Potencia y funcionalidad!\u00a1	2018	6136
Toyota	1888.615	Vendo mi veh\u00edculo elegante bajo consumo de combustible, historial de mantenimiento completo. \u00a1Una gran opci\u00f3n para cualquier conductor!	2011	10179
BMW	15693.185	Vendo mi SUV tracci\u00f3n en las cuatro ruedas, capacidad todoterreno, c\u00f3modo y espacioso. \u00a1La aventura te espera!	2008	38335

Anexo I: Pagina Web más información vehículo.

Figura 108

Página web más información vehículo.



RRochi

Pre Order

Land Rover 22499.74\$

Producto en Oferta

★★★★★

Auto compacto en excelente estado: bajo kilometraje, económico en combustible, ideal para la ciudad. ¡No te lo pierdas!

Punto de venta	Color
2546 Portage Place	Khaki
Kilometraje	Año
47426	2012

Contacto

Nombres
Anny Battison
Teléfono
391-215-3405
Dirección
276 Talisman Drive

Two smaller images showing the car from a rear perspective and a side view in an outdoor setting.

5.1 Resultados de la prueba de concepto

Para saber el porcentaje de eficiencia de la guía metodología se utilizó la medida de completitud funcional el cual mide la satisfacción de las necesidades del software, es el grado de las funcionalidades que cubren los objetivos específicos y las tareas planteadas por el usuario.

Para la evaluación de esta guía se realizó a través de la completitud funcional, consiste en medir la capacidad de la guía para realizar un sistema y el nivel de aceptación a nivel del usuario. Los aspectos para medir fueron las fases que se plantearon en la [sección 3.3](#) Ejecución de las fases Scrum, estas consisten en tres etapas: Fase 1 Pre-Juego, Fase 2 Juego y Fase 3 Post-Juego.

5.1.1. Análisis de resultados

Fórmula para medir la satisfacción en base a funciones

Para la medición de la guía se realizó con la ISO/IEC 25022 siguiendo la característica de la satisfacción el cual permitió medir la guía. La medida de funciones implica la medición de los objetivos, requisitos y los elementos específicos que se requieren evaluar proporcionados por el usuario, para tal caso es la guía metodológica.

Para medir la cobertura funcional, se plantea la pregunta de ¿Qué proporción de las funciones fueron satisfactorias? Mismo que se aplicó mediante una formula:

$$X = \frac{A}{B}$$

A: Número de Funciones satisfactorias

B: Número total de funciones

Nota: El resultado de la X se multiplicó por el valor de la escala de medición, la escala puede estar entre el rango cero y diez, para este caso se tomó como valor de medición el valor 10. Esto se realizó para saber en qué rango de puntuación se ubica el resultado.

Las funciones representan las tareas planteadas en la guía, es considerado función satisfactoria cuando cumple con el objetivo de la tarea.

Aplicación de la fórmula para la satisfacción de funciones

Cada una de las fases fueron puestas a prueba mediante la prueba de concepto, dentro de cada fase fueron propuestas diversas actividades, se encuentran en la [sección 2.2.1](#), en la Tabla 13 se ilustran los resultados:

Tabla 13

Evaluación de la guía mediante la completitud funcional.

Fase	Actividad	Tarea	Nivel de cumplimiento	Comentario u Observación
Pre-Juego	Sprint 0	Creación del proyecto, planeación.	SI	
		Roles	NO	únicamente se pudo agregar miembros al proyecto, pero no se pudo asignar roles
Juego	Sprint	Product Backlog	SI	
		Sprint Backlog	SI	

		Daily Scrum	SI	
		Sprint Review	SI	
		Retrospective	SI	
Post- Juego	Entrega	Increment	SI	
		Publisher	SI (50%)	Se despliega dependiendo del tipo de proyecto y del lenguaje de programación que se utiliza.

Dentro de la evaluación de la completitud funcional se encuentra 9 funciones que representan el 100%. Tras evaluación se obtuvo los siguientes resultados; 7,5 funciones que cumplen con los objetivos de la guía, contrario al 1,5 que no cumplen con las expectativas. El resultado tiene decimales porque existe una función que tiene limitaciones al cumplir el objetivo esperado. Porcentaje de éxito aplicando la función:

X: Porcentaje de éxito

A: Cantidad de funciones ausentes

B: Total de funciones

$$X = \frac{7,5}{9} = \mathbf{0,83}$$

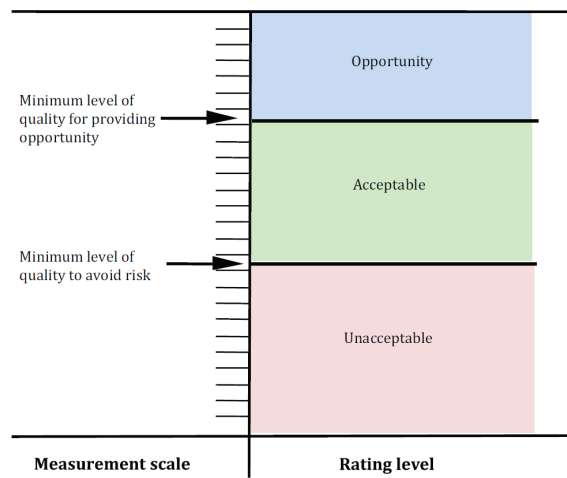
Con los resultados obtenidos tras la aplicación de la medición en base a funciones, se obtuvo un valor de 0,83. El cual indica que la calidad en uso es satisfactoria ya que se acerca al 1.

5.2.1 Evaluación de los resultados

La interpretación de los resultados se realizó en base a la escala de los niveles mínimos aceptables de calidad. La expresión de medición está determinando en términos de positivo o negativo tal como se observa en la siguiente imagen.

Figura 109

Niveles de calidad



Nota. Fuente (ISO/IEC 25022, 2016)

Para obtener el resultado del valor de medición de la guía metodológica se realizó el siguiente cálculo:

Valor resultante después de aplicar la fórmula de satisfacción de funciones multiplicado por la escala de medición esperado.

Escala de medición: 10

$$X = 0,83 * 10 = 8,3$$

Tras el resultado de 8,3 se pudo demostrar que la guía metodológica para la gestión de proyectos tiene un nivel de puntuación que cumple con los requisitos y el grado de satisfacción es oportuno.

Conclusiones

Luego de un análisis bibliográfico del marco de trabajo SCRUM y la herramienta GitHub se determinó la caracterización de la herramienta tecnológica GitHub, como también las especificaciones de las fases, roles, eventos, artefactos del marco de trabajo SCRUM que fueron la base teórica para fundamentar el desarrollo de la guía metodológica.

El desarrollo de la guía metodológica para la gestión de proyectos de software propone una herramienta para el desarrollo de proyectos de software ordenado y automatizado, la que consta de tres fases y varias tareas **Pre-Juego** "Creación y planeación del proyecto, roles, producto backlog", **Juego** "Sprint backlog, daily scrum, sprint review y retrospectiva" y **Post-Juego** "Increment and publish" con esto se pretende minimizar el tiempo y recurso para la gestión de proyecto.

En la evaluación de la guía metodológica propuesta se obtuvo un 83% de satisfacción en la utilización de las funciones desplegadas, es decir, obtuvo un grado de satisfacción oportuna y eficientemente para el desarrollo de proyectos de software basados en SCRUM y GitHub.

Recomendaciones

Para la aplicación de esta guía metodológica se recomienda tener conocimientos básicos del manejo de la herramienta Git y conocer la teoría del marco de trabajo SCRUM. También tener buena comunicación con los miembros del equipo.

Se recomienda que antes de aplicar esta guía se investigue cuáles son las nuevas actualizaciones implementadas en la herramienta de GitHub, con esto se evitara incoherencias en la ejecución de la guía metodológica. Para esto es recomendable leer la parte de las fases y tareas que están involucrados en la guía metodológica.

En caso de querer continuar con la investigación, se recomienda poner a prueba la guía con un proyecto de mayor magnitud y así descubrir cuáles son sus limitaciones, también se puede mejorar la guía apoyándose de otras herramientas o activando el plan de facturación.

Referencias

- Bhavsar, K., & Gopalan, S. (2020). Scrum: An Agile Process Reengineering in Software Engineering. *International Journal of Innovative Technology and Exploring Engineering*, 9(3), 840–848.
- Brent, B. (2018). *Introducing GitHub: Vol. Second Edition* (Kristen Brown). OREILLY.
- Business Review, H. (2017). *Gestión de proyectos*. Editorial Reverte. <https://elibro.net/es/lc/utnorte/titulos/46768>
- Coelho, J., Valente, M. T., Milen, L., & Silva, L. L. (2020). Is this GitHub project maintained? Measuring the level of maintenance activity of open-source projects. *Information and Software Technology*, 122, 106274. <https://doi.org/https://doi.org/10.1016/j.infsof.2020.106274>
- Davis, K. (2017). An empirical investigation into different stakeholder groups perception of project success. *International Journal of Project Management*, 35(4), 604–617. <https://doi.org/https://doi.org/10.1016/j.ijproman.2017.02.004>
- Diugwu, I. A., Mohammed, M., & Baba, D. L. (2015). Towards Effective Infrastructure Development in Nigeria: Theoretical Considerations from a Project Management Perspective. *American Journal of Industrial and Business Management*, 05(04), 172–180. <https://doi.org/10.4236/ajibm.2015.54019>
- El Bajta, M., & Idri, A. (2019). A Software Cost Estimation Taxonomy for Global Software Development Projects. *Proceedings of the 14th International Conference on Software Technologies*, 218–225. <https://doi.org/10.5220/0007841202180225>
- El Bajta, M., Idri, A., Ros, J. N., Fernandez-Aleman, J. L., Carrillo de Gea, J. M., Garcia, F., & Toval, A. (2018). Software project management approaches for global software development: a systematic mapping study. *Tsinghua Science and Technology*, 23(6), 690–714. <https://doi.org/10.26599/TST.2018.9010029>
- Espinoza Mina, A. M., & Gallegos Barzola, D. del P. (2017). La industria del software en Ecuador: evolución y situación actual. *Pág*, 38, 25. <https://www.revistaespacios.com/a17v38n57/a17v38n57p25.pdf>
- Gonçalves, L. (2018). Scrum. *Controlling & Management Review*, 62(4), 40–42. <https://doi.org/10.1007/s12176-018-0020-3>
- Guía del PMBOK®. (2008). *Guía de los Fundamentos para la Dirección de Proyectos* (Cuarta Edición). Project Management Institute, Inc.
- Hadida, S., & Troilo, F. (2020, October). La agilidad en las organizaciones: Trabajo comparativo entre metodologías ágiles y de cascada en un contexto de ambigüedad y transformación digital. *ECONSTOR*. <http://hdl.handle.net/10419/238381>

- ISO/IEC 25022. (2016). *Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use*. www.iso.org
- Kertesz, C.-Z. (2015). Using GitHub in the classroom - a collaborative learning experience. *2015 IEEE 21st International Symposium for Design and Technology in Electronic Packaging (SIITME)*, 381–386. <https://doi.org/10.1109/SIITME.2015.7342358>
- KPMG. (2013). *Project management survey report*.
- Kuz, A., Falco, M., & Giandini, R. S. (2018, June 10). Comprendiendo la Aplicabilidad de Scrum en el Aula: Herramientas y Ejemplos. *Revista Iberoamericana de Tecnología En Educación y Educación En Tecnología*, 21, 62–70.
- Lamprou, A., & Vagiona, D. (2018). Success criteria and critical success factors in project success: a literature review. In *INTERNATIONAL JOURNAL OF REAL ESTATE AND LAND PLANNING* (Vol. 1).
- López, D. S., Ramírez Pérez, J. F., & Gutiérrez Fera, L. M. (2017, March). La integración en la gestión de proyectos: diagnóstico y buenas prácticas a implementar en la UCI. *Serie Científica*, 55–71.
- Modelo, E., & Sanchez Orduña, R. (2018). *Guía Práctica PM4* (BID-INDES, Ed.). BID-INDES.
- Monte Galiano, J. (2016). *Implantar scrum con éxito*. Editorial UOC. <https://elibro.net/es/lc/utnorte/titulos/58575>
- Morandini, M., Coleti, T. A., Oliveira, E., & Corrêa, P. L. P. (2021). Considerations about the efficiency and sufficiency of the utilization of the Scrum methodology: A survey for analyzing results for development teams. *Computer Science Review*, 39, 100314. <https://doi.org/https://doi.org/10.1016/j.cosrev.2020.100314>
- Moreno Perez, J. C. (2015). *Administracion de software de un sistema informatico*. RA-MA Editorial. <https://elibro.net/es/lc/utnorte/titulos/62503>
- ODS. (2022). *objetivos de Desarrollo Sostenible*. Naciones Unidas.
- Project Management Institute. (2017). *Agile Practice Guide* (Project Management Institute, Ed.). Project Management Institute.
- Ravishankar, S. (2013). *Git: Version Control for Everyone Beginner's Guide* (First Edition). Packt Publishing Ltd.
- Reina Guaña, E. P., Patiño Rosado, S. G., & Quijosaca, F. (2019). Evaluación de la calidad en uso de un sistema web/ móvil de control de asistencia a clases de docentes y estudiantes aplicando la norma ISO/IEC 25000 SQuaRe. *Risti*.
- Robles Belmonte, M. T. (2017). *Guía Metodológica*. <https://docplayer.es/user/38697302/>
- Rodriguez Moreno, D. C. (2020). *La Industria del software en Boyaca*. Editorial UPTC. <https://elibro.net/es/lc/utnorte/titulos/193942>

- Romero-Romero, A., & López-Botello, F. Y. (2020, September 25). Acompañamiento Docente en Proyectos Informáticos de Desarrollo de Software para el Usuario Final en una Institución de Educación Superior. *Revista Tecnológica-Educativa Docentes 2.0*, 9(2), 212–222. <https://doi.org/https://doi.org/10.37843/rted.v9i2.166>
- Sarmiento Rojas, J. A., Correa Candamil, C. H., & Jimenez Roa, D. E. (2020). *Gestión de proyectos aplicada al PMBOK 6ED*. Editorial UPTC. <https://elibro.net/es/lc/utnorte/titulos/193943>
- Schwaber, K., & Sutherland, J. (2020). *Manifiesto for Agile Software Development*.
- Shafiq, M., Zhang, Q., Akbar, M. A., Khan, A. A., Hussain, S., Amin, F.-E., Khan, A., & Soofi, A. A. (2018). Effect of Project Management in Requirements Engineering and Requirements Change Management Processes for Global Software Development. *IEEE Access*, 6, 25747–25763. <https://doi.org/10.1109/ACCESS.2018.2834473>
- Terlizzi, M. A., Meirelles, F. de S., & de Moraes, H. R. O. C. (2016). Barriers to the use of an IT Project Management Methodology in a large financial institution. *International Journal of Project Management*, 34(3), 467–479. <https://doi.org/https://doi.org/10.1016/j.ijproman.2015.12.005>
- The Stationery Office, & Lea, W. (2017). *Managing Successful Projects with PRINCE2®: Vol. Sixth edition (AXELOS)*. TSO.
- Toor, S.-R., & Ogunlana, S. O. (2010). Beyond the ‘iron triangle’: Stakeholder perception of key performance indicators (KPIs) for large-scale public sector development projects. *International Journal of Project Management*, 28(3), 228–236. <https://doi.org/https://doi.org/10.1016/j.ijproman.2009.05.005>
- Velmurugan, M. (2021, January 28). *GIT FOR NOOB*. CLASS ADVISER. <https://classadviser.in/git/>
- Wattanakriengkrai, S., Chinthanet, B., Hata, H., Kula, R. G., Treude, C., Guo, J., & Matsumoto, K. (2022). GitHub repositories with links to academic papers: Public access, traceability, and evolution. *Journal of Systems and Software*, 183, 111117. <https://doi.org/https://doi.org/10.1016/j.jss.2021.111117>
- Zagalsky, A., Feliciano, J., Storey, M.-A., Zhao, Y., & Wang, W. (2015). The Emergence of GitHub as a Collaborative Platform for Education. *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 1906–1917. <https://doi.org/10.1145/2675133.2675284>
- Zakiah, A., & Fauzan, M. N. (2016). Collaborative Learning Model of Software Engineering using Github for informatics student. *2016 4th International Conference on Cyber and IT Service Management*, 1–5. <https://doi.org/10.1109/CITSM.2016.7577521>
- Zapata Gomez, A. (2015). *Ciclo de la calidad PHVA*. Editorial Universidad Nacional de Colombia. <https://elibro.net/es/lc/utnorte/titulos/129837>

Zöller, N., Morgan, J. H., & Schröder, T. (2020). A topology of groups: What GitHub can tell us about online collaboration. *Technological Forecasting and Social Change*, 161, 120291. <https://doi.org/https://doi.org/10.1016/j.techfore.2020.120291>

ANEXOS

GUÍA PARA LA GESTIÓN DE PROYECTOS DE SOFTWARE CON LA HERRAMIENTA GITHUB Y MARCO DE TRABAJO SCRUM

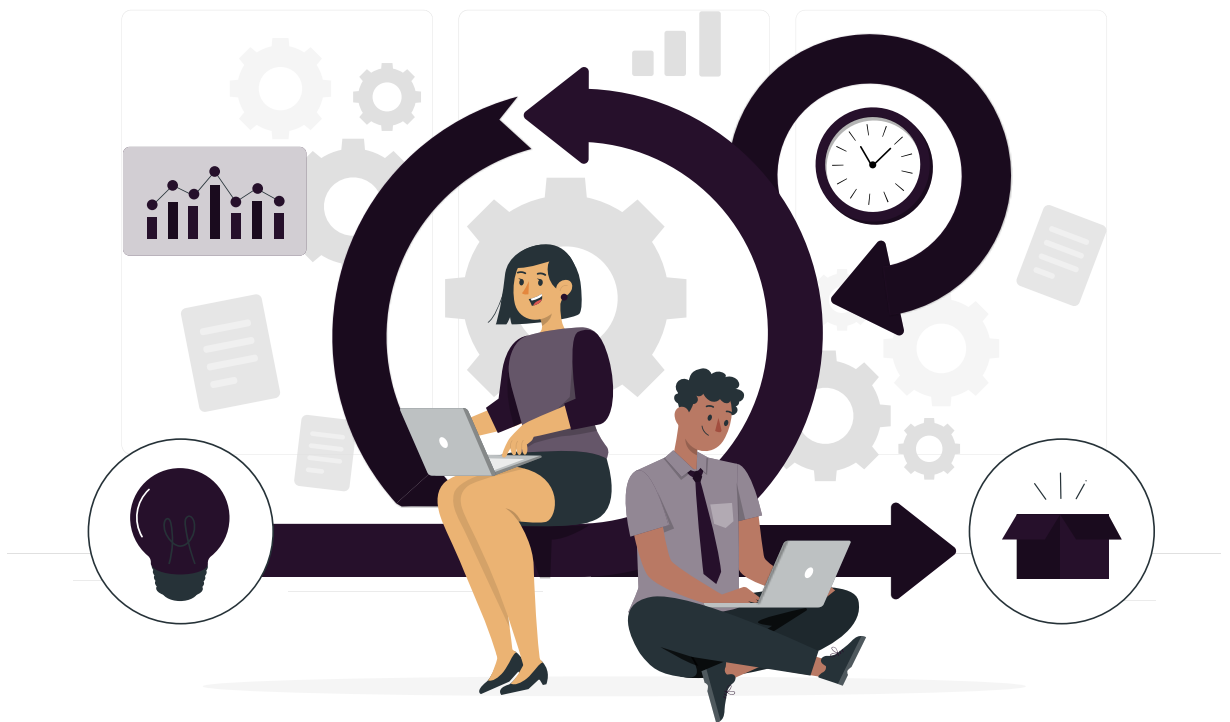


TABLA DE CONTENIDO

INTRODUCCIÓN

DESARROLLO

CONCLUSIONES

INTRODUCCIÓN

Este documento presenta una guía metodológica para la gestión de proyectos, tiene como objetivo orientar a los desarrolladores de software en el proceso del desarrollo de software apoyándose del marco de trabajo SCRUM y la herramienta GITHUB.

Dentro del documento se detalla las instrucciones de como iniciar la gestión de proyectos hasta culminar con la entrega del producto o servicio.

DESARROLLO

Es primordial conocer cuáles serán los miembros que conformarán el grupo de trabajo, cantidad de integrantes no debe exceder de las 10 personas recomendadas por el marco de trabajo SCRUM. Cada uno de los miembros debe tener asignado un rol.

El guía desarrollado se encuentra dividida en tres fases, inicia con Pre-Juego dentro de esta fase se planifica cuáles son las actividades para realizar, las actividades son los requerimientos de los usuarios.

La segunda fase es el Juego dentro de esta fase se desarrolla y planifica los Sprint “conjunto de actividades”.

Finalizando con la fase de Post-Juego, el cual consiste en realizar entregables o el producto final al cliente.

A. Configuración del entorno GITHUB

Dentro de este apartado se crea y configura el proyecto para la gestión del desarrollo de software. También se agrega a los miembros del equipo al proyecto para tener un breve acercamiento hacia el uso de la herramienta.

Crear cuenta de GITHUB

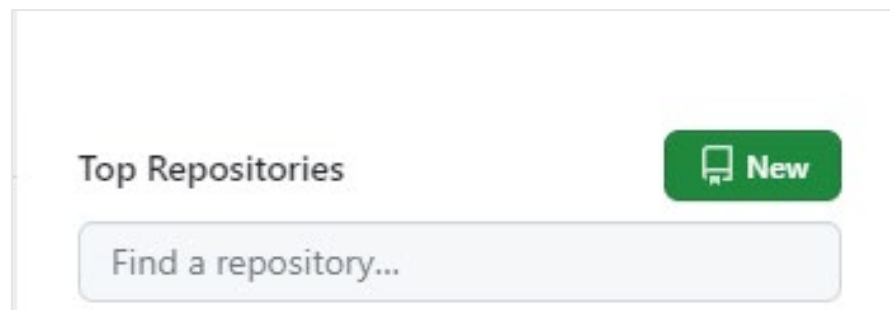
1. Acceder al siguiente enlace <https://github.com/> dar clic en **sign in > Create an account**
2. Ingrese los datos solicitados en la plataforma GITHUB.

Crear y agregar miembros al proyecto

1. Dentro de la cuenta GITHUB dar clic en **new**

Ilustración 1

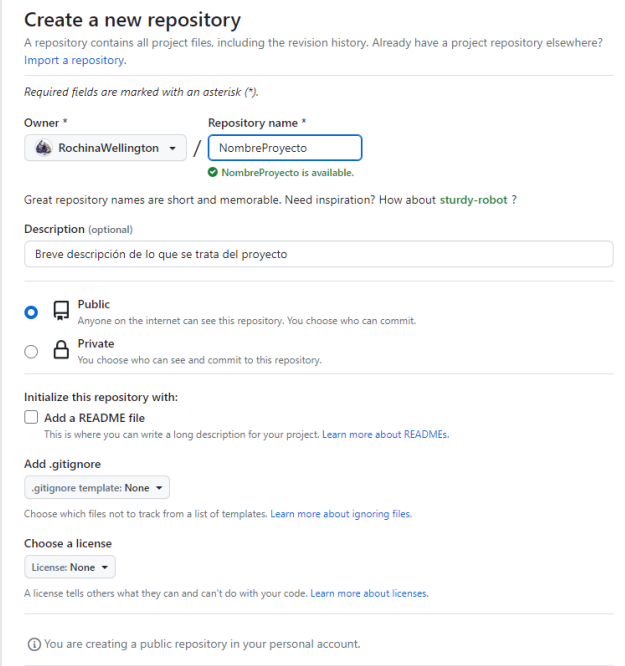
Crear Repositorio.



2. Ingreso los datos que requiera para la creación del proyecto “nombre, descripción, tipo público o privado”

Ilustración 2

Ingreso de información para crear repositorio.

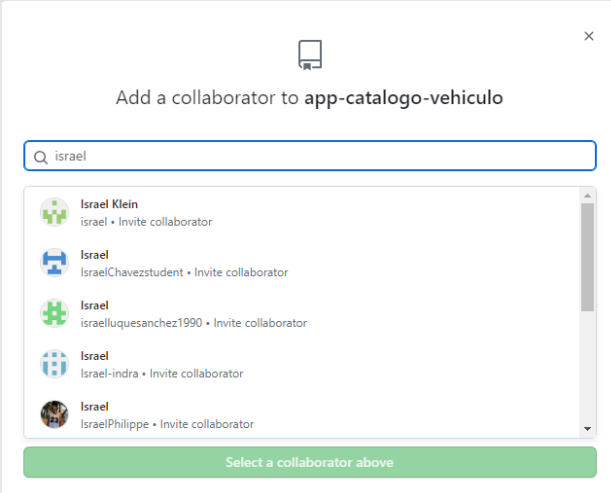


The screenshot shows the 'Create a new repository' page. At the top, it says 'Create a new repository' and explains that a repository contains all project files. Below this, there are fields for 'Owner' (set to 'RochinaWellington') and 'Repository name' (set to 'NombreProyecto'). A green checkmark indicates that 'NombreProyecto' is available. There is a text area for 'Description (optional)' with the placeholder text 'Breve descripción de lo que se trata del proyecto'. Below the description, there are radio buttons for 'Public' (selected) and 'Private'. Underneath, there are options to 'Initialize this repository with:' including 'Add a README file' and 'Add .gitignore' (with a dropdown menu set to 'None'). There is also a 'Choose a license' dropdown menu set to 'None'. At the bottom, a note states 'You are creating a public repository in your personal account.'

3. Agregar miembros, clic en **settings > collaborators > Add people** ingrese el mail o nombre completo y seleccione el integrante.

Ilustración 3

Agregar miembros al repositorio.



The screenshot shows a dialog box titled 'Add a collaborator to app-catalogo-vehiculo'. It features a search bar with the text 'israel'. Below the search bar, there is a list of search results, each showing a profile picture, the name 'Israel', and the text 'Invite collaborator'. The results include 'Israel Klein', 'IsraelChavezstudent', 'Israel', 'Israel-indra', and 'IsraelPhilippe'. At the bottom of the dialog, there is a green button that says 'Select a collaborator above'.

Crear un nuevo repositorio o subir uno existente

1. Para crear un nuevo proyecto, tener instalado git en la maquina local y ejecutar los siguientes comandos que son proporcionados por GITHUB.

Ilustración 4

Crear nuevo repositorio con comandos desde la maquina local.

...or create a new repository on the command line

```
echo "# ProyectoTest" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/RochinaWellington/ProyectoTest.git
git push -u origin main
```

2. Para subir un proyecto existente al proyecto creado ejecutar los siguientes comandos proporcionados por la herramienta GITHUB.

Ilustración 5

Subir el proyecto local al repositorio nuevo.

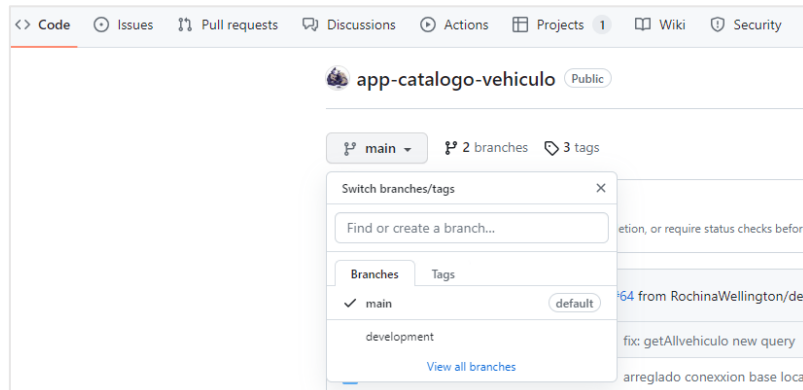
...or push an existing repository from the command line

```
git remote add origin https://github.com/RochinaWellington/ProyectoTest.git
git branch -M main
git push -u origin main
```

3. Dado el caso de crear o subir un repositorio en el apartado de **<>Code** se mostrará las ramas y los archivos existentes.

Ilustración 6

Visualizar proyecto subido desde la maquina local.



B. PRE-JUEGO

En esta fase se debe tener definido cual es el propósito o producto para desarrollar. La fase está dividida en 3 etapas creación del proyecto, asignación de roles y Product Backlog.

Ilustración 7

Pre-Juego.

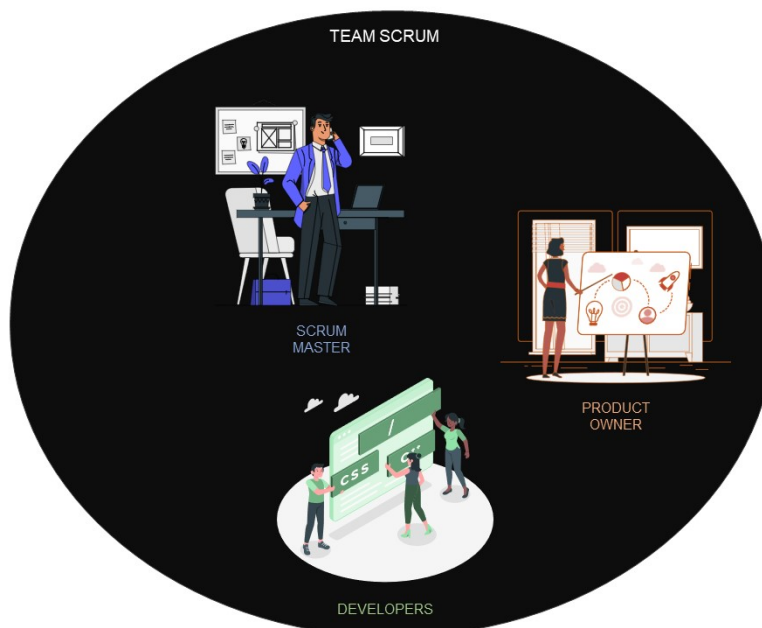


Roles

En la *Ilustración 8* se observan los roles del equipo son personas que forman parte del desarrollo de un proyecto, aportan con sus conocimientos, pueden ser: Product Owner “Dueño del producto”, Scrum Master “Líder de proyecto” y Developers “Colaboradores en el desarrollo”. Tener en cuenta; dentro del grupo no existe jerarquías. Los miembros del equipo se autogestionan por sí mismas es decir que son capaces de tomar decisiones y cumplir con el objetivo de obtener un producto (Schwaber & Sutherland, 2020).

Ilustración 8

Roles en Scrum



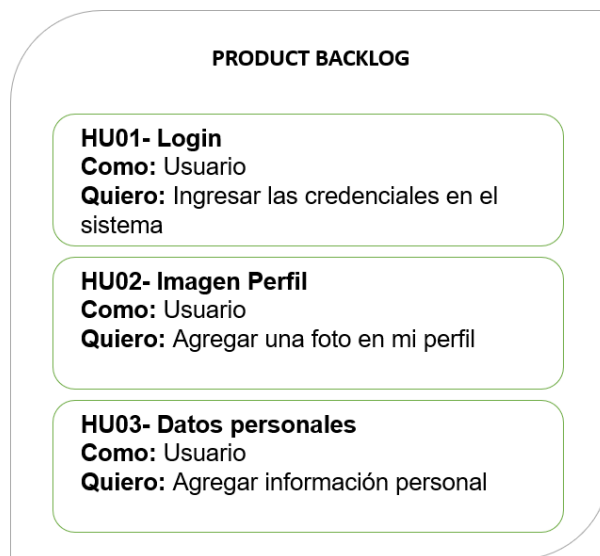
Product Backlog

Como se observa en la Ilustración 9, son requerimientos definidos por producto owner que se realizan durante antes de iniciar a desarrollar el software, el producto Backlog contiene historia de usuarios. La lista de producto tiene una descripción, estimación, orden y valor (Gonçalves, 2018). La persona encargada de tener la lista de requerimientos es el Scrum Master.

Los elementos de un Product Backlog son realizados por el equipo de trabajo, listo para que se tomen en la planificación de un evento de Sprint. Los desarrolladores involucrados en la creación del Product Backlog son responsables del dimensionamiento, también pueden estar involucrado el Product Owner ayudando a comprender y seleccionar compensaciones

Ilustración 9

Product backlog.



Creación del proyecto, planeación.

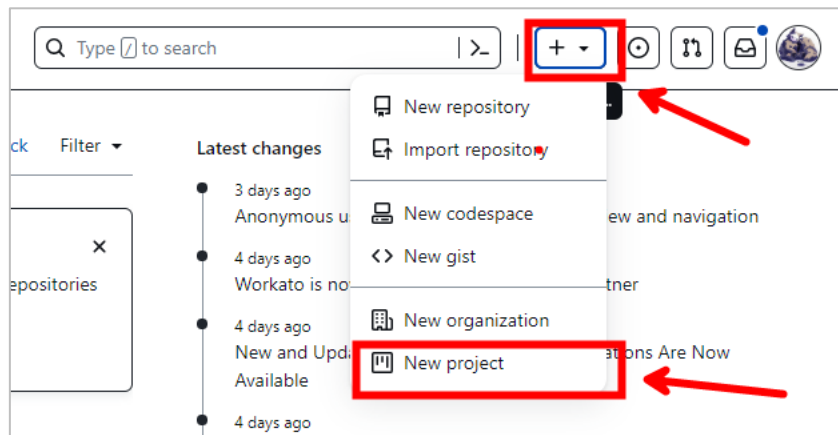
Los pasos por seguir para crear un nuevo proyecto dentro de la herramienta GitHub son:

Todos los miembros del equipo deben estar al tanto del propósito del proyecto para así distribuir los roles en base a sus habilidades y experiencias.

1. Una vez dentro de la cuenta, dar clic en el botón con icono de **+**, tras esto se despliega una lista de opciones, seleccionar **new Project**.

Ilustración 10

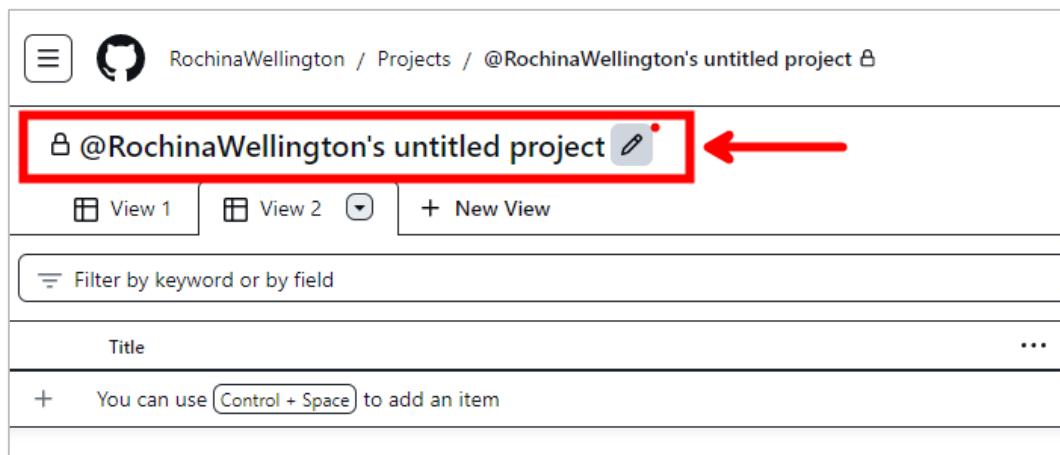
Crear nuevo proyecto.



2. Dentro del proyecto creado puede renombrar el nombre del proyecto que se crea por defecto. Dar clic en el signo del lápiz o sobre el nombre.

Ilustración 11

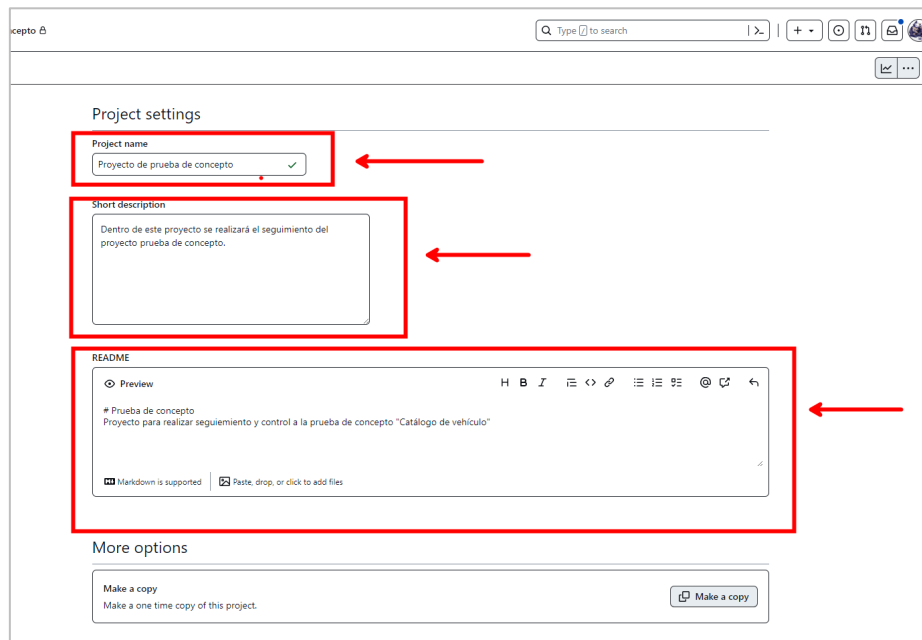
Editar nombre del proyecto.



3. Ingrese los datos que desee editar y guarde los cambios.

Ilustración 12

Ingreso de información para editar proyecto.



Asignación de Roles.

GitHub en su plan libre no permite la asignación de Roles, por lo que se lo debe hacer con una herramienta externa en el que se tenga anotado cuáles serán los roles de cada miembro del equipo.

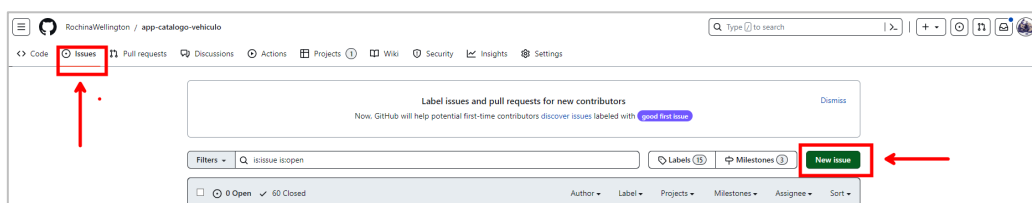
Product Backlog

Dentro de GitHub puede crear lista Product Backlog "Lista de tareas". Estos son los requerimientos para cumplir con el objetivo del proyecto, suelen ser necesidades del cliente.

1. Desde la página de inicio, ingrese al repositorio que creó dentro dar clic en **issues** >> **new issue**.

Ilustración 13

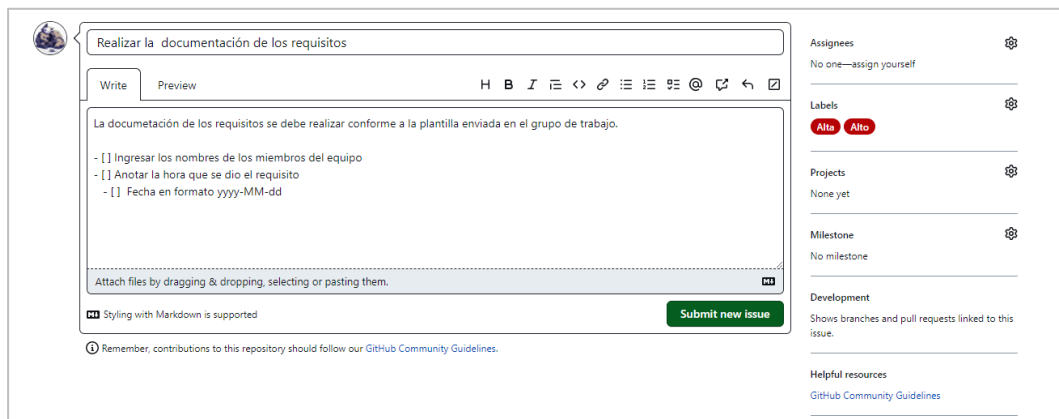
Crear issue.



- Ingreso los datos solicitados. Cada issue puede ser etiquetada por el nivel de dificultad o la urgencia de hacer la tarea en el apartado de labels. También puede asignar la tarea a una persona, aunque esto se debe asignar más adelante.

Ilustración 14

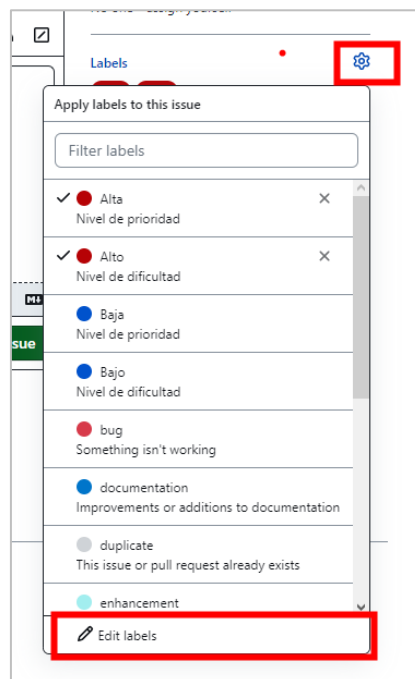
Ingresar información a la issue.



- 2.1. Puede crear más labels de las que viene por defecto, clic en el icono de la tuerca y edit labels.

Ilustración 15

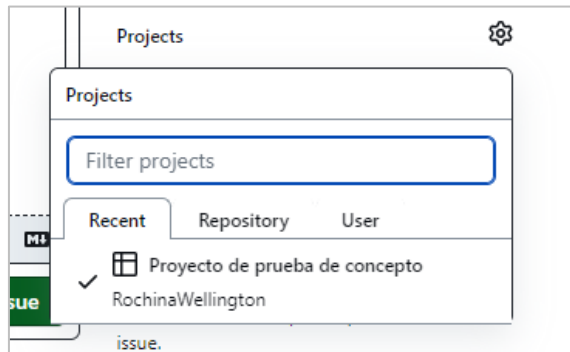
Editar etiquetas.



- 2.2. Es importante asignar la tarea al proyecto que se creó con antelación. Clic en projects.

Ilustración 16

Agregar issue al proyecto.



3. Para visualizar que la tarea se ha asignado al proyecto, clic en **Projects** e ingrese al proyecto correspondiente.

Ilustración 17

Visualizar proyecto.

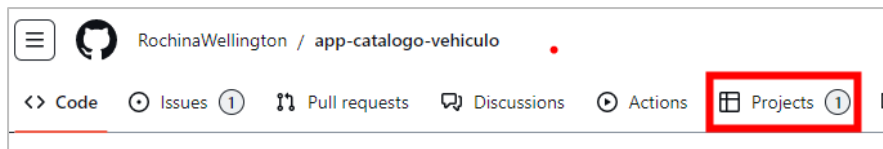


Ilustración 18

Gestión de proyecto.

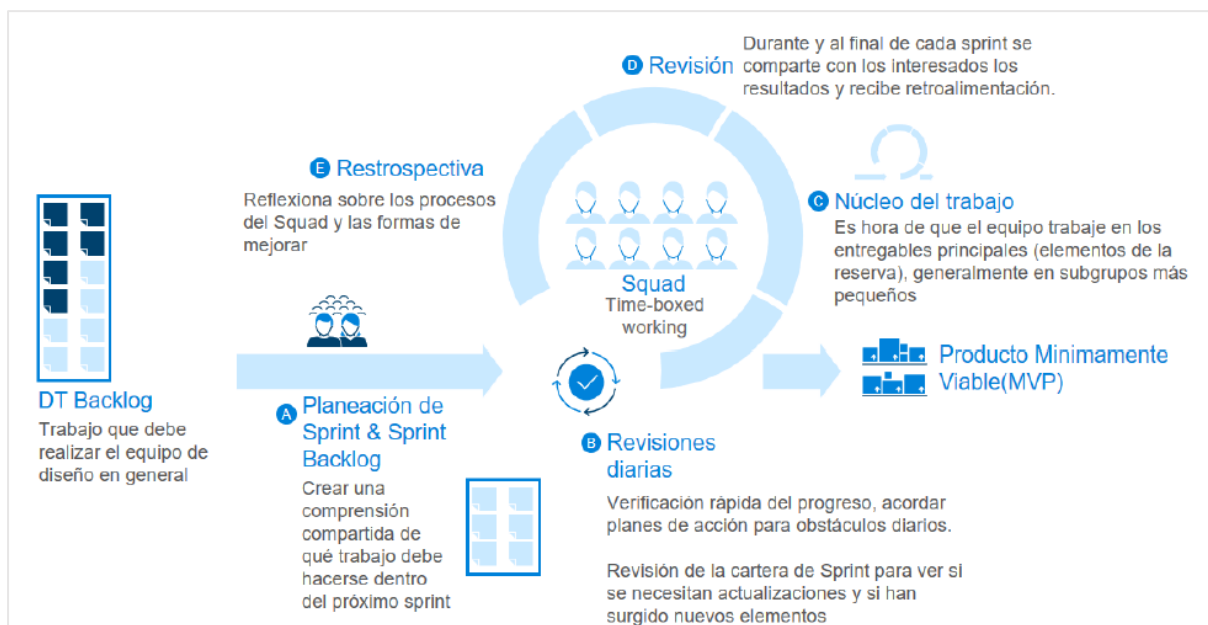
Title	Assignees	Status	Estim...	Linked pull requests	Labels	Iteration
1 Realizar el entregable de la aplicación #25	israelrochin...	Done	8		Alta	Iteration 4
2 Mostrar la información de acerca de nosotros (redes sociales) #1	israelrochina	Done	2		Bajo	Iteration 3
3 Consultar el contacto de los vehículos #2	israelrochina	Done	2		Alta	Iteration 3
4 Consultar más información del vehículo en venta #3	israelrochina	Done	2		Alto	Iteration 3
5 Consulta de vehículos en venta #4	israelrochina	Done	2		Media	Iteration 3
6 Realizar el consumo del api de usuario/contacto #24	israelrochina	Done	3		Media	Iteration 3
7 Realizar el consumo del api de tipo de vehículo #23	israelrochina	Done	3		Bajo	Iteration 3
8 Realizar la conexión al servicio de backend #18	israelrochina	Done	1		Bajo	Iteration 3
9 Realizar el consumo de modelo de vehículo #22	israelrochina	Done	3		Media	Iteration 3
10 Realizar el consumo de la api de modelo #21	israelrochina	Done	1		Alto	Iteration 3
11 Realizar el consumo del api de vehículos #20	israelrochina	Done	2		Alto	Iteration 3
12 realizar el consumo de la api de Usuarios #19	israelrochina	Done	2		Alto	Iteration 3
13 Configuración de backend con servidor de express #10	israelrochina	Done	2		Alto	Iteration 2
14 Realizar conexión a la BD #11	israelrochina	Done	1		Alto	Iteration 2

C. JUEGO

Dentro de esta fase se distribuye y ejecuta las tareas. La fase está dividida en 5 etapas Sprint Backlog, Daily Scrum, Sprint Review y Retrospective.

Ilustración 19

Juego.



Nota. Fuente (Hadida & Troilo, 2020).

Sprint Backlog

El Sprint backlog son listas de tareas del Product Backlog, están organizados y categorizados por el nivel de dificultad, los sprint tienen límite de tiempo para ser completados por el Team Scrum. Generalmente de un Sprint anterior se puede realizar mejoras al Sprint Backlog según como se vaya aprendiendo, esto puede ayudar en la inspección del progreso en Daily Scrum.

Daily Scrum

Daily Scrum involucra la revisión de trabajos de manera diaria, con el objetivo de solventar dificultades que tienen los miembros del equipo para completar la lista de tareas del Sprint. Ayuda a mejorar la comunicación identificar impedimentos, promueve la toma de decisiones rápido y elimina las necesidades de las reuniones anteriores. El Daily Scrum no es el único momento en que los desarrolladores pueden reunirse, también pueden reunirse a lo largo del día para posibles inquietudes o pedir más detalles del Sprint (Schwaber & Sutherland, 2020).

Sprint Review

Una vez que se haya concluido con el Sprint, es necesario realizar un Sprint Review. Es atendido por el Scrum Team y Stakeholders quien es invitado por el Product Owner. El Sprint Review tiene una duración aproximada de 4 horas en el caso de que el Sprint dure un mes. Se realiza una revisión para detectar que es lo que se ha logrado durante el Sprint, revisión de las tareas de Sprint Review anterior.

Después de Sprint Review el Product Backlog puede ser reajustado para satisfacer nuevas necesidades. Se revisan los elementos probables del Product Backlog para el próximo Sprint (Schwaber & Sutherland, 2020).

Sprint Retrospective

El objetivo de Sprint Retrospective es planear un incremento de calidad y eficacia. El Team Scrum investiga cuáles fueron los resultados del último Sprint con respecto a las personas, los procesos, las iteraciones, las herramientas. Estas inspecciones pueden variar dependiendo del dominio del trabajo. Se exploran cuáles fueron los inconvenientes que llevaron a un mal camino, qué dificultades tuvieron, identifica los cambios más útiles que mejoraron su efectividad. Las mejoras que tuvieron buenos resultados se pueden incorporar al Sprint Backlog.

El Sprint Retrospective tiene una duración de 3 horas en el caso de que el Sprint haya durado un mes, en otros Sprint cortos requiere una reunión corta. Este Sprint Retrospective es esencial para centrarse en inspección, adaptación y mejoramiento (Gonçalves, 2018).

Sprint Backlog “Iteraciones”.

Esta etapa parte del Producto Backlog. Con el fin de realizar las tareas, se agrupa las tareas por iteraciones. Las iteraciones tienen un límite de tiempo de una semana para ser desarrolladas, cada tarea dentro de las iteraciones es asignado a los miembros del equipo.

Los pasos para seguir para crear iteraciones y asignar tareas son:

1. Dentro del proyecto creado hacer clic sobre el nombre del proyecto.

Ilustración 20

Acceder al proyecto para crear iteraciones.



2. Dentro clic en **Iteration >> more options**. Seleccionar la fecha inicio y escoger la duración en semanas, ingrese los números de semanas que durara en terminar el proyecto.

Ilustración 21

Crear iteraciones.

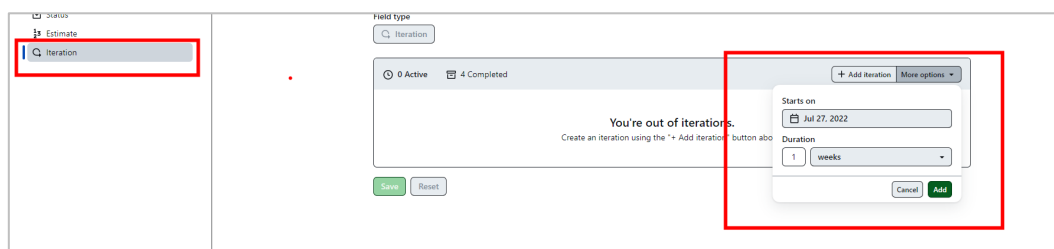


Ilustración 22

Visualizar iteraciones.

0 Active	4 Completed	
Completed	Iteration 1 1 week Jun 29, 2022 - Jul 05, 2022	🗑️
Completed	Iteration 2 1 week Jul 06, 2022 - Jul 12, 2022	🗑️
Completed	Iteration 3 1 week Jul 13, 2022 - Jul 19, 2022	🗑️
Completed	Iteration 4 1 week Jul 20, 2022 - Jul 26, 2022	🗑️

3. Asignar tareas del proyecto a una de las iteraciones conforme al nivel de urgencia que se deba realizar.
 - 3.1. Clic en el signo + para agregar la columna de iteración.

Ilustración 23

Crear columna de iteración.

The screenshot shows a Jira project board for 'Proyecto de prueba de concepto'. The board has columns for 'Home', 'Current iteration', 'Next iteration', 'Planning', and 'Iteration 1'. A task list is displayed with columns for 'Title', 'Assignees', and 'Status'. A red box highlights a '+' icon in the header row, and a red arrow points to it. Another red box highlights the 'Iteration' option in the field configuration menu, with a red arrow pointing to it.

Title	Assignees	Status	+
1 Realizar el entregable de la aplicación #25	israelrochin...	Done	+ New field
2 Mostrar la información de acerca de nosotros (redes sociales) #1	israelrochina	Done	Visible fields
3 Consultar el contacto de los vehículos #2	israelrochina	Done	<input checked="" type="checkbox"/> Title
4 Consultar más información del vehículo en venta #3	israelrochina	Done	<input checked="" type="checkbox"/> Assignees
5 Consulta de vehículos en venta #4	israelrochina	Done	<input checked="" type="checkbox"/> Status
6 Realizar el consumo del api de usuario/contacto #24	israelrochina	Done	Hidden fields
7 Realizar el consumo del api de tipo de vehículo #23	israelrochina	Done	<input type="checkbox"/> Labels
8 Realizar la conexión al servicio de backend #18	israelrochina	Done	<input type="checkbox"/> Linked pull requests
9 Realizar el consumo de modelo de vehículo #22	israelrochina	Done	<input type="checkbox"/> Reviewers
10 Realizar el consumo de la api de modelo #21	israelrochina	Done	<input type="checkbox"/> Repository
11 Realizar el consumo del api de vehículos #20	israelrochina	Done	<input type="checkbox"/> Milestone
12 realizar el consumo de la api de Usuarios #19	israelrochina	Done	<input type="checkbox"/> Estimate
13 Configuración de backend con servidor de express #10	israelrochina	Done	<input type="checkbox"/> Iteration
14 Realizar conexión a la BD #11	israelrochina	Done	

- 3.2. Asignar tareas a las iteraciones.

Ilustración 24

Asignar tareas a la iteración.

Title	Assignees	Status	Iteration
1 Realizar el entregable de la aplicación #25	israelrochin...	Done	Iteration 4
2 Mostrar la información de acerca de nosotros (redes sociales) #1	israelrochina	Done	Iteration 3
3 Consultar el contacto de los vehículos #2	israelrochina	Done	Iteration 3
4 Consultar más información del vehículo en venta #3	israelrochina	Done	Iteration 3
5 Consulta de vehículos en venta #4	israelrochina	Done	Iteration 3
6 Realizar el consumo del api de usuario/contacto #24	israelrochina	Done	Iteration 3
7 Realizar el consumo del api de tipo de vehículo #23	israelrochina	Done	Iteration 3
8 Realizar la conexión al servicio de backend #18	israelrochina	Done	Iteration 3
9 Realizar el consumo de modelo de vehículo #22	israelrochina	Done	Iteration 3
10 Realizar el consumo de la api de modelo #21	israelrochina	Done	Iteration 3
11 Realizar el consumo del api de vehículos #20	israelrochina	Done	Iteration 3

4. Asignar responsable a las tareas.

Ilustración 25

Asignar responsable a la tarea.

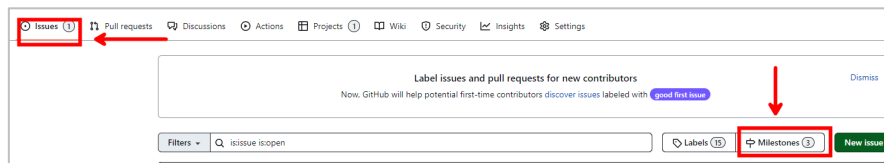
Title	Assignees	Status	Iteration
1 Realizar el entregable de la aplicación #25	israelrochin...	Done	Iteration 4
2 Mostrar la información de acerca de nosotros (redes sociales) #1	israelrochina	Done	Iteration 3
3 Consultar el contacto de los vehículos #2	israelrochina	Done	Iteration 3
4 Consultar más información del vehículo en venta #3	israelrochina	Done	Iteration 3
5 Consulta de vehículos en venta #4	israelrochina	Done	Iteration 3
6 Realizar el consumo del api de usuario/contacto #24	israelrochina	Done	Iteration 3
7 Realizar el consumo del api de tipo de vehículo #23	israelrochina	Done	Iteration 3
8 Realizar la conexión al servicio de backend #18	israelrochina	Done	Iteration 3

5. Una vez asignado las tareas, los responsables de cada tarea pueden iniciar a realizar sus actividades. Deben clonar el repositorio en sus máquinas locales y crear una rama para trabajar con el fin de no dañar la rama principal **main**.

- 5.1. Para clonar el repositorio en la maquina local, clic en **code** copiar el enlace "HTTPS, SSH o GitHub CLI". En la maquina local crear una carpeta y abrir con git para clonar el repositorio.

Ilustración 28

Crear milestone.



2. Clic en New milestone e ingrese los datos requeridos para crear.

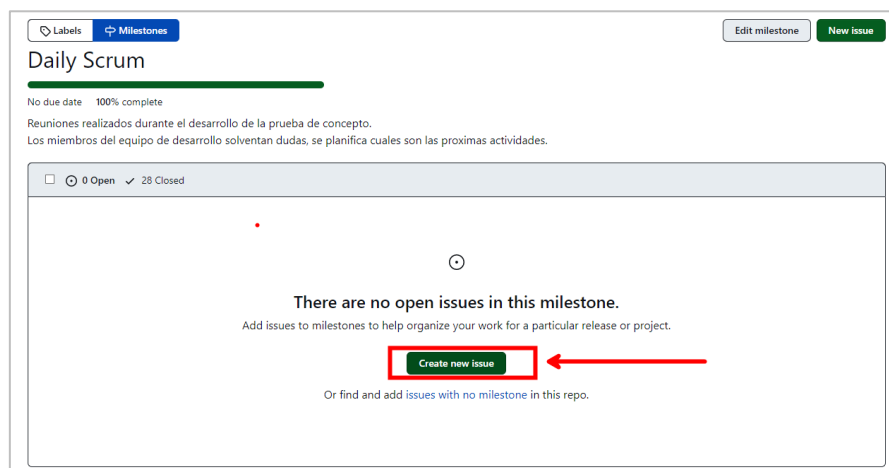
Ilustración 29

Ingresar información para crear milestone.

3. Crear las reuniones, clic en create new issue dentro del milestone creado.

Ilustración 30

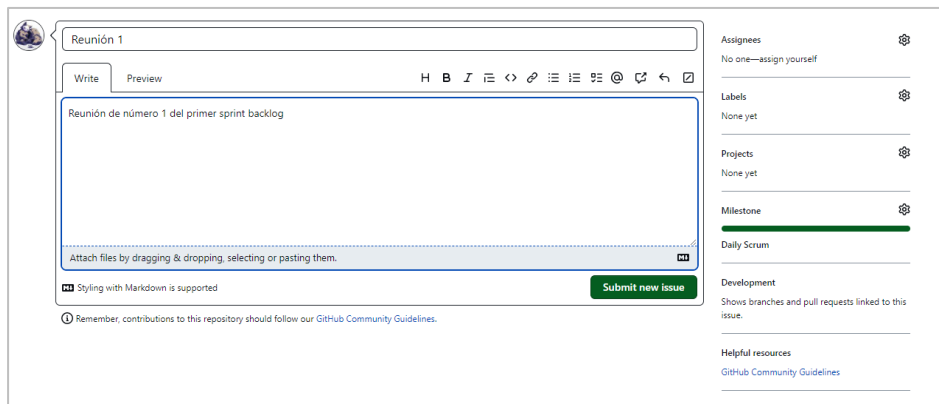
Crear issue dentro de milestone.



3.1. Ingreso los datos necesarios para crear el issue.

Ilustración 31

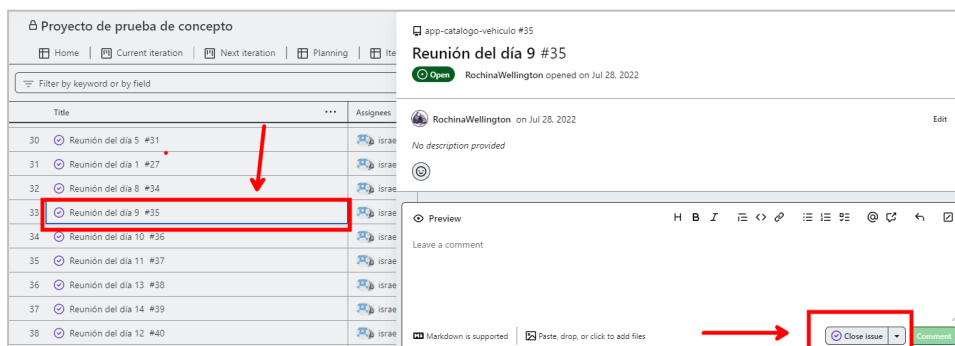
Ingreso información para crear issue.



4. Para dar por finalizado una reunión, dentro del proyecto hacer clic sobre la tarea y dar clic en Close issue.

Ilustración 32

Dar por finalizado una reunión "issue".



COMMITTS

5. Para monitorizar y revisar los avances de las tares puede hacerse con los **commits**. Puede visualizar los commits por cada rama creado a partir de las tareas "issue".

5.1. Seleccione la rama que desea monitorizar, clic en commits.

Ilustración 33

Ingresar a los commits.

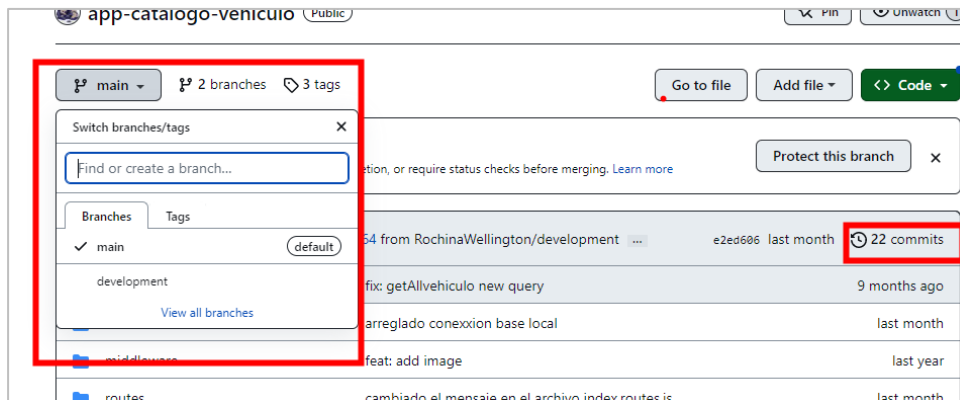
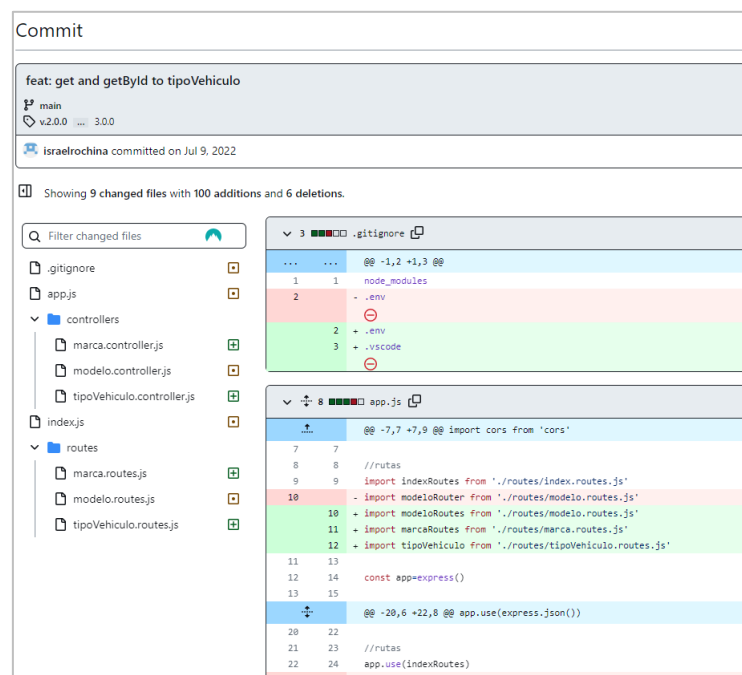


Ilustración 34

Visualizar cambios del commit.



Sprint Review

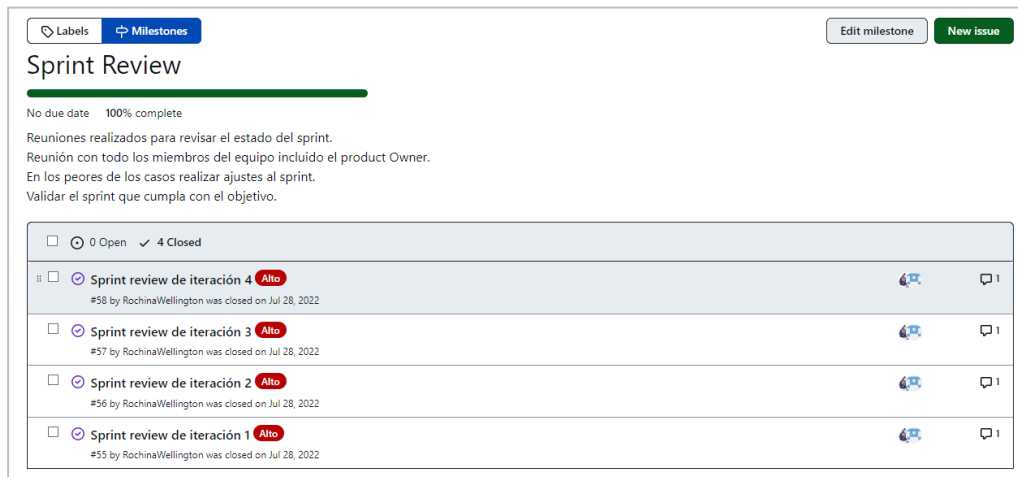
Una vez finalizado la iteración es necesario revisar si se logró culminar con las tareas que se planificaron. Los responsables de las tareas mencionan los inconvenientes que han tenido y el scrum master junto al producto owner determinan si la tarea se da como finalizado. En caso de existir comentarios acerca de la actividad esta es registrado.

Nota: Esto se hace por cada iteración finalizado.

Para registrar y organizar los sprint review se hace crear un milestone. Los pasos para crear el milestone y issue son los mismos que el de daily scrum.

Ilustración 35

Gestion sprint review.



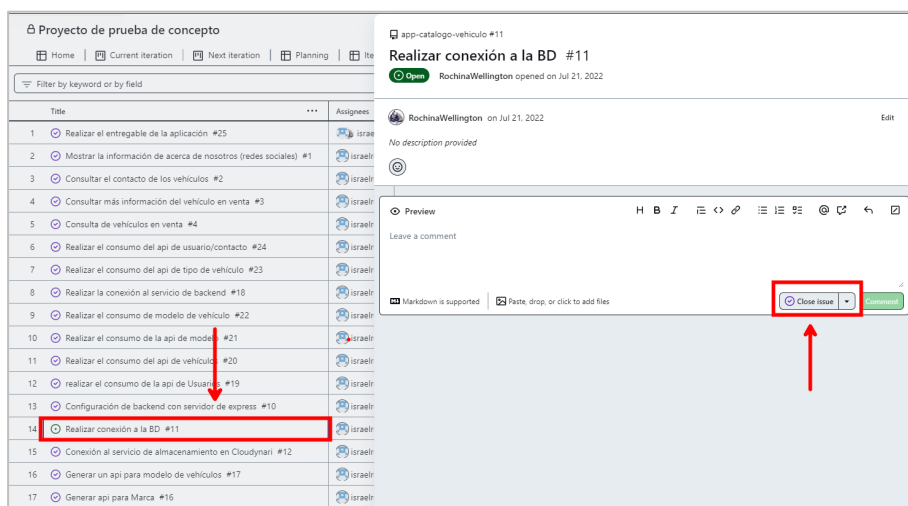
FINALIZAR TAREAS

Para finalizar tareas se puede hacer de dos maneras:

1. Acceder al proyecto, seleccionar tarea y dar clic en close issue.

Ilustración 36

Finalizar tarea issue.



2. En caso de haber creado una rama a partir de la tarea. Desde la maquina local con la consola de git puede ejecutar el siguiente comando.

Ilustración 37

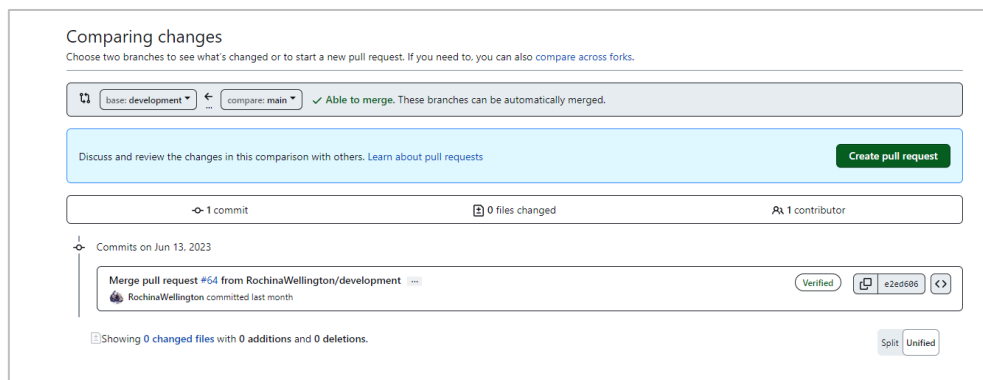
Finalizar issue por comando.

```
$ git commit -m "Close #id_del_issue"
```

- 2.1. Una vez que los cambios estén subidos en la rama de la tarea hacer un pull request hacia la rama main o rama donde se esté integrando todos los cambios. Después de hacer pull request elimine la rama de la tarea para no tener exceso de ramas no utilizadas.

Ilustración 38

Crear pull request.



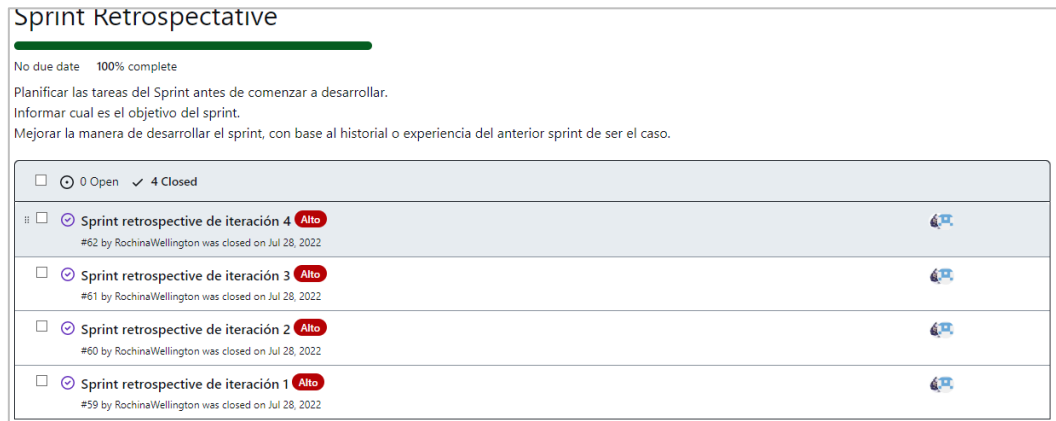
Retrospective

La retrospective se realiza con el fin de anotar cuales fueron las dificultades y como se solucionaron los problemas al realizar una tarea. Con el fin de mejorar y no cometer los mismos errores al realizar la próxima iteración.

Para registrar y organizar la retrospective crear el milestone e issue correspondiente. Los pasos para seguir son los mismos que en Daily Scrum.

Ilustración 39

Gestión retrospectiva.

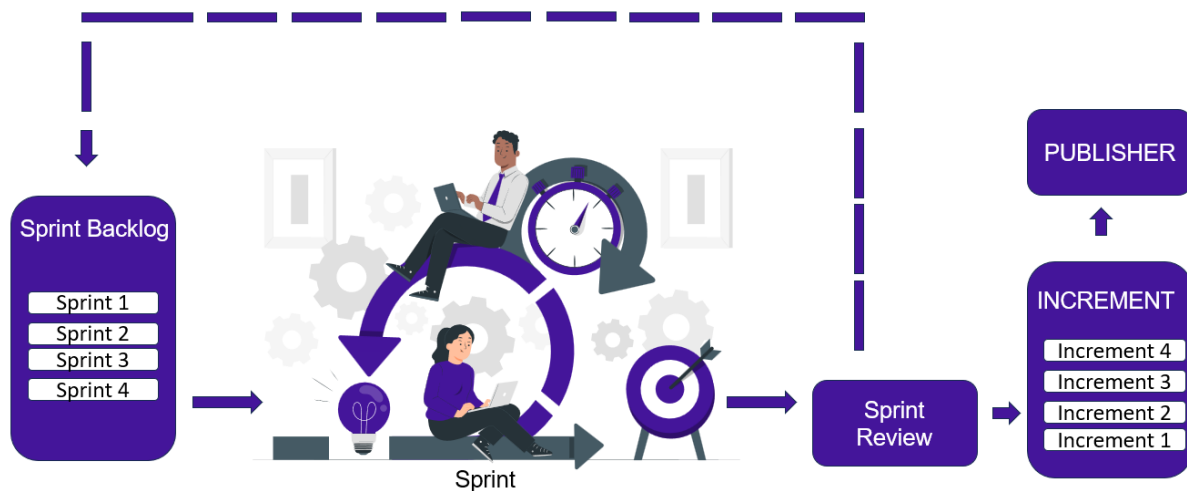


D. POST – JUEGO

Esta fase está dividida en 2 etapas Increment y Publisher.

Ilustración 40

Post-Juego.



Increment

Se define como incremento cuando se da por finalizado el Sprint y cumple con la necesidad del cliente, el incremento se obtiene de cada Sprint Backlog y es el acumulado de Sprint, se considera como respaldos. Los incrementos pueden ser entregados a la parte interesada para que pueda ver los avances.

Publisher

El publisher, es la entrega final del software. Se obtiene del último incremento ya que este contiene el código final de todas las tareas que se han planteado al inicio del proyecto.

Increment

Después de finalizar una iteración es necesario guardar los cambios para prevenir cualquier cambio no deseado y se dañe los archivos.

1. Los incrementos se realizan con la función de Tag. Para crear un se debe hacer desde la consola como se mira en la figura **Error! Reference source not found.**

Ilustración 41

Crear Tag.

```
$ git tag -a "nombre de la etiqueta" -m "mensaje de la etiqueta"
```

2. Subir el tag al repositorio.

Ilustración 42

Subir tag al repositorio.

```
$ git push origin "nombre de la etiqueta"
```

3. Los tags se visualizan haciendo clic en tags.

Ilustración 43

Visualizar tags.



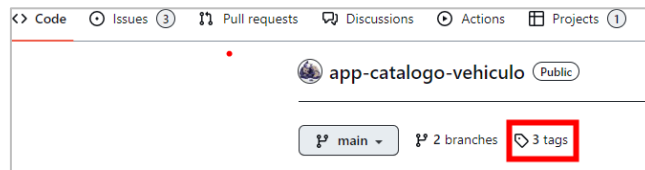
Publisher

Finalizado el proyecto se debe crear un **release**, contiene código final del proyecto. El archivo release está en dos formatos **zip y tar.gz** son de este formato para que no pueda ser editado solo descargado. Pasos para seguir.

1. Dentro del repositorio clic en **tags**.

Ilustración 44

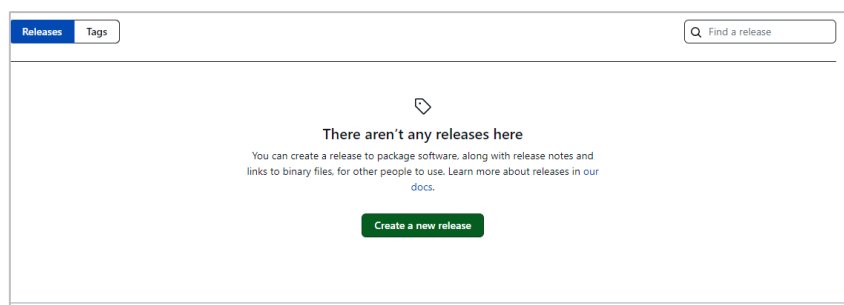
Acceder a los release.



2. Clic en **Releases >> Create a new release.**

Ilustración 45

Crear release.



3. Un release se crea a partir de un tag. Seleccione el tag, ingrese el nombre y puede incluir una descripción.

Ilustración 46

Ingreso de información para crear release.

3.0.0

✓ Existing tag

Entregable del proyecto

Write Preview

H B I Previous tag: auto Generate release notes

Entregable del proyecto
- Este el entregable del proyecto que fue probado con el cliente.

Attach files by dragging & dropping, selecting or pasting them.

↓ Attach binaries by dropping them here or selecting them.

Set as a pre-release
This release will be labeled as non-production ready

Set as the latest release
This release is labeled as the latest for this repository.

Update release

CONCLUSIONES

La guía se desarrolló haciendo uso de dos herramientas, GitHub y el marco de trabajo SCRUM. Por lo cual es necesario tener conocimiento del manejo de GitHub sobre todo los comandos de Git para trabajar desde la maquina local.

La guía es intuitiva de utilizar, contiene ilustraciones de cómo crear issues, milestone, Project y Branch. Es de utilidad para las personas que desean gestionar sus proyectos sobre todo trabajar en equipo.