

Integrantes:

Juan José Afanador Ochoa

Stiven González Olaya

Santiago Roa Dueñas

Gabriel Andrés Niño Carvajal

John Jairo González Martínez

*Buen Vecino*

Pontificia Universidad Javeriana

Ingeniería de Sistemas

Ingeniería de Software



**Docente:** Carlos Andrés Parra Acevedo

Versión 2.0

# Historial de cambios

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Encargado |
| 24/08/2020 | 0.1 | Creación documento, modificación canvas y prototipado de aplicación | Juan Afanador  Santiago Roa |
| 24/08/2020 | 0.2 | Elaboración diagrama de Gantt | Juan Afanador, Gabriel Niño |
| 24/08/2020 | 0.3 | Propósito, alcances y objetivos | Gabriel Niño |
| 24/08/2020 | 0.4 | Modelo de ciclo de vida  Lenguajes y herramientas | Juan Afanador  Stiven Gonzalez |
| 25/08/2020 | 0.5 | Resumen, supuestos y restricciones | Juan Afanador |
| 25/08/2020 | 0.6 | Ambiente de trabajo | Santiago Roa |
| 25/08/2020 | 0.7 | Lenguajes y herramientas, Visión del producto | Stiven Gonzalez y John Gonzalez |
| 26/08/2020 | 0.8 | Entregables, Análisis y administración de riesgos | John Gonzalez |
| 26/08/2020 | 0.9 | Administración de configuración y documentación | Stiven Gonzalez |
| 27/08/2020 | 0.10 | Plan de aceptación del producto | Gabriel Niño y John Gonzalez |
| 27/08/2020 | 0.11 | Organización del proyecto y comunicación | Juan Afanador y Santiago Roa |
| 28/08/2020 | 0.12 | Inicio del proyecto | Stiven Gonzalez y Juan Afanador |
| 01/09/2020 | 0.13 | Métodos y herramientas de estimación | Santiago Roa |
| 02/09/2020 | 0.14 | Planes de trabajo del proyecto | Juan Afanador |
| 02/09/2020 | 0.15 | Administración de requisitos | Gabriel Niño |
| 05/09/2020 | 0.16 | Monitoreo y control del progreso, control de calidad, glosario | Gabriel Niño y Stiven Gonzalez |
| 07/09/2020 | 0.17 | Cierre del proyecto | Gabriel Niño |
| 08/09/2020 | 0.18 | Lista de tablas, lista de figuras, tabla de contenidos | Todos los miembros del grupo. |
| 08/09/2020 | 0.19 | Entrega del producto | John Gonzalez |
| 09/09/2020 | 1.0 | Revisión final documento | Todos los miembros del grupo. |
| 05/10/2020 | 1.01 | Resumen | Santiago Roa |
| 05/10/2020 | 1.02 | Propósito, alcance y objetivos | Gabriel Niño |
| 05/10/2020 | 1.03 | Supuestos y restricciones | Gabriel Niño |
| 05/10/2020 | 1.04 | Glosario, Lenguajes y herramientas | John Gonzalez |
| 05/10/2020 | 1.05 | Organización del proyecto y comunicación | Juan Afanador |
| 06/10/2020 | 1.06 | Métodos y herramientas de estimación | Santiago Roa, Juan Afanador |
| 06/10/2020 | 1.07 | Administración de requisitos, Cierre del proyecto | Gabriel Niño, John Gonzalez |
| 06/10/2020 | 1.08 | Análisis y administración de riesgos | Gabriel Niño, Santiago Roa |
| 06/10/2020 | 1.09 | Control de calidad | Stiven Gonzalez, |
| 06/10/2020 | 2.0 | Historial de cambios | Juan Afanador |

Tabla 1 Historial de cambios

# Resumen

En el documento se da a conocer el proyecto BuenVecino, el cual es desarrollado por el grupo ECO, conformado por estudiantes de la Pontificia Universidad Javeriana. El aplicativo busca enfocarse en la población universitaria la cual carece de una residencia permanente y busca hospedarse en algún lugar de acuerdo con sus necesidades y locación especifica.

Es necesario tener en cuenta todas las necesidades y objetivos del proyecto, por esto mismo en el documento se especifican todos los requisitos planteados por el docente para poder llevar a cabo el proyecto de la mejor manera, por medio de una constante comunicación y control de procesos.

Para poder llevar a cabo este proyecto es necesario una excelente gestión del tiempo, herramientas de administración y control que permitan realizar avances periódicos significativos, adicional a ello es altamente eficiente el establecer un flujo de trabajo y un modelo de ciclo de vida los cuales recaigan en la idea y contexto de la idea.

Este proyecto se desarrolla con el objetivo de crear interés en aquellos que deseen entender el proceso de trabajo del grupo Eco, sus metodologías y herramientas usadas para poder crear el proyecto BuenVecino y el paso a paso de la producción partiendo desde la idea.

# Tabla de contenidos

[1 Historial de cambios 2](#_Toc50534639)

[2 Resumen 3](#_Toc50534640)

[3 Tabla de contenidos 4](#_Toc50534641)

[4 Lista de figuras 6](#_Toc50534642)

[5 Lista de tablas 6](#_Toc50534643)

[6 Vista general del proyecto 7](#_Toc50534644)

[6.1 Visión del producto 7](#_Toc50534645)

[6.2 Propósito, alcance y objetivos 7](#_Toc50534646)

[6.3 Supuestos y restricciones 8](#_Toc50534647)

[6.4 Entregables 8](#_Toc50534648)

[6.5 Evolución del plan 9](#_Toc50534649)

[6.6 Glosario 10](#_Toc50534650)

[7 Contexto del proyecto 11](#_Toc50534651)

[7.1 Modelo de ciclo de vida 11](#_Toc50534652)

[7.1.1 Metodologías agiles: *Scrum* 11](#_Toc50534653)

[7.1.2 *Extreme* *programming* 12](#_Toc50534654)

[7.1.3 Kanban 13](#_Toc50534655)

[7.1.4 Metodología seleccionada 13](#_Toc50534656)

[7.2 Lenguajes y herramientas 14](#_Toc50534657)

[7.2.1 Herramientas y lenguajes de desarrollo 14](#_Toc50534658)

[7.2.2 Desarrollo Web 14](#_Toc50534659)

[7.2.3 Herramientas de control de versiones y hosting de versiones 17](#_Toc50534660)

[7.2.4 Herramientas de Modelado 18](#_Toc50534661)

[7.2.5 Manejo de documentos 19](#_Toc50534662)

[7.2.6 Manejo del proyecto 20](#_Toc50534663)

[7.3 Plan de aceptación del producto 20](#_Toc50534664)

[7.3.1 Entregables 20](#_Toc50534665)

[7.4 Organización del proyecto y comunicación 21](#_Toc50534666)

[7.4.1 Interfaces externas o *stakeholder* 21](#_Toc50534667)

[7.4.2 Organigrama y descripción de roles 22](#_Toc50534668)

[8 Administración del proyecto 27](#_Toc50534669)

[8.1 Métodos y herramientas de Estimación 27](#_Toc50534670)

[8.1.1 *Planning* *Poker* 27](#_Toc50534671)

[8.2 Inicio del proyecto 29](#_Toc50534672)

[8.2.1 Plan de capacitación 29](#_Toc50534673)

[8.2.2 Infraestructura del proyecto 30](#_Toc50534674)

[8.2.3 Tareas para realizar 32](#_Toc50534675)

[8.3 Planes de trabajo del proyecto 32](#_Toc50534676)

[8.3.1 Estructura de descomposición de tareas (WBS) 32](#_Toc50534677)

[8.3.2 Calendarización del proyecto 32](#_Toc50534678)

[8.3.3 Presupuesto del proyecto 33](#_Toc50534679)

[9 Monitoreo y control del proyecto 34](#_Toc50534680)

[9.1 Administración de requisitos 34](#_Toc50534681)

[9.2 Monitoreo y control del progreso 34](#_Toc50534682)

[9.2.1 Medidas del proyecto 34](#_Toc50534683)

[9.2.2 Actividades para realizar 35](#_Toc50534684)

[9.2.3 Acciones correctivas 35](#_Toc50534685)

[9.3 Cierre del proyecto 36](#_Toc50534686)

[10 Entrega del producto 39](#_Toc50534687)

[11 Procesos de soporte 40](#_Toc50534688)

[11.1 Ambiente de trabajo 40](#_Toc50534689)

[11.1.1 Reglas de trabajo 40](#_Toc50534690)

[11.1.2 Excusas válidas 40](#_Toc50534691)

[11.1.3 Sanciones y acciones correctivas 41](#_Toc50534692)

[11.2 Análisis y administración de riesgos 41](#_Toc50534693)

[11.2.1 Proceso de identificación de riesgos 41](#_Toc50534694)

[11.2.2 Clasificación de riesgos 41](#_Toc50534695)

[11.2.3 Tratamiento del riesgo 42](#_Toc50534696)

[11.2.4 Riesgos identificados 42](#_Toc50534697)

[11.3 Administración de configuración y documentación 43](#_Toc50534698)

[11.4 Control de calidad 44](#_Toc50534699)

[11.4.1 Medidas de Calidad: 44](#_Toc50534700)

[11.4.2 Proceso para control de calidad de los documentos 45](#_Toc50534701)

[11.4.3 Proceso para control de calidad de software: 45](#_Toc50534702)

[12 Anexos 47](#_Toc50534703)

[13 Referencias 48](#_Toc50534704)

# Lista de figuras

[Figura 1 Modelo de ciclo de vida ECO 14](#_Toc50574221)

[Figura 2 Organigrama equipo de trabajo ECO 23](#_Toc50574222)

[Figura 3 Planning Poker Sprint 28](#_Toc50574223)

[Figura 6 Diagrama de Gantt 34](#_Toc50574224)

[Figura 7 Diagrama BPMN Modificar Requisitos 35](#_Toc50574225)

[Figura 8 Diagrama BPMN Identificación de Riesgos 42](#_Toc50574226)

[Figura 9 Diagrama BPMN control de cambios 45](#_Toc50574227)

[Figura 10 Diagrama BPMN control de calidad de la documentación 46](#_Toc50574228)

[Figura 11 Diagrama BPMN control de calidad del Software 47](#_Toc50574229)

# Lista de tablas

[Tabla 1 Historial de cambios 2](#_Toc50542906)

[Tabla 2 Entregables del proyecto 9](#_Toc50542907)

[Tabla 3 Glosario 10](#_Toc50542908)

[Tabla 4 Entregables con criterios y herramientas 21](#_Toc50542909)

[Tabla 5 Skateholders 22](#_Toc50542910)

[Tabla 6 Descripción de roles 27](#_Toc50542911)

[Tabla 7 Herramientas de Software 31](#_Toc50542912)

[Tabla 8 Infraestructura de Hardware 32](#_Toc50542913)

[Tabla 9 Acciones correctivas 36](#_Toc50542914)

[Tabla 10 Ítems de configuración 44](#_Toc50542915)

# Vista general del proyecto

## Visión del producto

Al ser Bogotá la capital del país posee una importancia alta en diferentes aspectos sociales, como lo son el trabajo y la educación. Respecto a esta última, a la ciudad llega una enorme cantidad de personas en busca de educación universitaria. Esto ocasiona que el costo general de vida en la ciudad aumente, lo que aumenta de forma considerable los costos relacionados a vivienda, siendo esta uno de los aspectos más importantes a la hora de migrar en busca de educación hacía la ciudad. Por lo anteriormente mencionado, muchos universitarios buscan opciones temporales de vivienda que se ajusten a sus necesidades económicas y que se encuentren lo más cercanas posible a la universidad donde estudiarán.

El principal objetivo de BuenVecino, es ser intermediario entre espacios de arrendamiento y estudiantes de todas partes del país, ofreciendo lugares para hospedarse mientras estudian, dichos lugares deben contar con las comodidades deseadas por el cliente, todo esto mediante una plataforma ágil y segura en la cual se garantice una buena experiencia de hospedaje.

.

## Propósito, alcance y objetivos

* **Propósito:** El proyecto BuenVecino se va a realizar con el objetivo de ofrecer una oportunidad rentable para aquellas personas que dispongan de un espacio extra en sus hogares para alquilarlo por un determinado periodo de tiempo. BuenVecino quiere invitar a todo aquel estudiante universitario cuya necesidad sea la de buscar un sitio agradable y cómodo para hospedarse por el tiempo que sea necesario.
* **Alcance:** Nuestro proyecto estará desarrollado en plataforma web netamente enfocada en estudiantes universitarios de la ciudad de Bogotá. Además, se planea contar con una interfaz amigable tanto para nuestros socios como para nuestros clientes, esto es en busca de crear buena relación con los usuarios para obtener una mayor satisfacción de estos. Esta plataforma web contará con un chat de mensajes entre usuarios (arrendador y arrendatario), sin embargo, no contará con ningún tipo de soporte técnico debido al alcance del proyecto. Será indispensable que la plataforma de BuenVecino cuente con un espacio para efectuar transacciones monetarias. BuenVecino contará con tres tipos de suscripción para nuestros socios clave, los *arrendatarios*. Los tipos de suscripción que se manejaran son de tipo mensual, trimestral y anual.
* **Objetivo General:** Ofrecer una solución de vivienda temporal exclusivamente para estudiantes universitarios y una oportunidad económica para aquellas personas que cuenten con un espacio disponible en casa para compartir.
* **Objetivos Específicos:** 
  + Guardar estructuradamente datos tanto de usuario como de vivienda
  + Aplicar conceptos y metodologías vistas en Ingeniería de *software*
  + Enviar notificaciones periódicas a los usuarios del sistema.
  + Investigar sobre métodos de *geolocalización* e integración web y entregar información geográfica visual
  + Generar pasarela de pago.

## Supuestos y restricciones

**Supuestos:**

* Los tiempos de entrega deben cumplirse de manera obligatoria siguiendo la *diagramación* *Gantt*.
* Los roles de grupo están establecidos de manera uniforme y con cargas de trabajo equitativas.
* La base de datos utilizada para el proyecto será implementada únicamente bajo la plataforma *Firebase*.
* El usuario debe contar con un equipo de cómputo o celular para acceder a la plataforma del proyecto.
* Es necesario que el usuario cuente con una conexión a internet mientras usa la plataforma.
* El usuario debe estar dispuesto a registrarse en la plataforma para poder realizar una reserva.

**Restricciones:**

* El tiempo de planeación y desarrollo corresponde al segundo semestre académico de 2020, los intervalos de tiempo para desarrollar los *Sprint* deben ser respetados para dar cabida a las presentaciones orales.
* El proyecto únicamente funcionará como una plataforma Web, en cualquier dispositivo que pueda conectarse a una red de internet.
* Para ejecutar el proyecto el usuario debe contar con un navegador compatible con *JavaScript*
* Solo se garantiza el correcto despliegue de la parte grafica (*frontend*) en el navegador Google Chrome.
* La plataforma únicamente estará disponible en Bogotá.

## Entregables

|  |  |  |  |
| --- | --- | --- | --- |
| **Entregable** | **Descripción** | **Dirigido a** | **Fecha límite de entrega** |
| SPMP | Documento con lo referente a la administración, planeación, vista general y contexto del proyecto. | Profesor | 09/09/2020 |
| SRS | Documento con lo referente a la especificación de los requisitos del sistema | Profesor | 16/10/2020 |
| Primer Prototipo | Prototipo funcional con la implementación del caso de uso más difícil. | Profesor | 16/10/2020 |
| SDD | Documento con la descripción del diseño del software | Profesor | Por definir |
| Segundo Prototipo | Prototipo funcional con al menos el 70% de los casos de uso implementados | Profesor | Por definir |

Tabla 2 Entregables del proyecto

## Evolución del plan

A lo largo de las distintas reuniones es muy probable que se identifiquen posibles factores o puntos a modificar, estos deben ser modificados siguiendo una metodología que permita realizar dichos cambios sin alterar la funcionalidad general del proyecto, es decir de manera modular. Para ello frente a la necesidad de generar cualquier tipo de cambio se deben seguir los siguientes pasos:

1. Identificar punto a modificar: Cualquier miembro del grupo dentro del *sprint* es capaz de identificar posibles pivotes con respecto a algún modulo del aplicativo, esta idea debe ser comunicada en la próxima reunión del grupo y los demás integrantes tendrán la posibilidad de analizar dicha perspectiva y de ser aprobado el cambio se realizará la identificación de oportunidades y posible implementación.
2. Reunión de cambio: Si en definitiva se considera como positivo el cambio sugerido se procederá a realizar una reunión grupal para acordar el cambio y realizar la respectiva documentación, esta reunión podrá realizarse en cualquiera de las reuniones planteadas semanalmente o bien de ser necesario la reunión podrá realizarse de acuerdo con el horario de los participantes para lograr un avance significativo.
3. Identificación de impacto: En la reunión se evaluará el impacto posible que pueda generar la implementación de la idea, esto con respecto al tiempo y recursos a utilizar con el objetivo de seguir la metodología *scrum*.
4. Implementación: Al final de la reunión se procede a llevar a cabo el plan de cambio elaborado para modificar los respectivos numerales del proyecto, sus variables y su funcionalidad de una manera segura y sostenible.
5. Evaluación posterior: Se busca identificar el impacto y sus efectos positivos y negativos dentro del proyecto, tomando como punto de análisis la funcionalidad original.
6. Control de versiones: Todo esto debe ser realizado mediante alguna herramienta de control de cambios, en este caso se utilizará *GitHub* para medir el cambio a través del tiempo, con respecto a la documentación, esta será guardada en la carpeta compartida del grupo para su correcta verificación y posible modificación.

## Glosario

|  |  |
| --- | --- |
| Termino | Definición |
| Arrendatario | Aquel individuo que toma una determinada cosa o servicio en arrendamiento. [1] |
| Arrendamiento | Es la cesión, adquisición o aprovechamiento temporal, ya sea de cosas, obras, servicios, o cambio de un valor.[2] |
| Inmueble | Se refiere a algo que está unido al terreno de modo inseparable tanto física como jurídicamente, en este caso se utilizará para referirse a apartamentos, casas o habitaciones. [3] |
| Apartamento | Vivienda que integra un edificio en el cual hay otras viviendas similares. [4] |
| Casa | Se refiere a una construcción de una o pocas plantas que está destinada a la vivienda de una única familia. [5] |
| Habitación | Lugar que se establece mediante paredes en el interior de una construcción. [6] |
| Amoblado | Participio de amoblar, significa poner, dotar o equipar con muebles. [7] |
| Contrato | Acuerdo escrito, por el que dos o más partes de comprometen a respetar y cumplir una serie de condiciones. [8] |
| Fiador | Persona que se compromete a garantizar la ejecución de un contrato por una de las partes en provecho de la otra. [9] |
| Geolocalización | Consiste en obtener la ubicación geográfica de un objeto. Esto podría ser un *smartphone*, un automóvil, una calle, etc. [10] |
| Stakeholder | Se refiere a las partes interesadas, en el ámbito empresarial, y aquellas personas que se verían afectadas por las decisiones de una empresa.[11] |
| Frameworks | Es una estructura conceptual y tecnología de soporte definido, muy útil para la organización y desarrollo de software.[12] |
| Sprint | Ciclos o iteraciones que vamos a tener dentro de un proyecto *Scrum*. [13] |
| Hosting | Servicio en línea que permite un sitio o una aplicación en internet. [14] |
| Librería | Es un conjunto de implementaciones funcionales, codificadas en un lenguaje de programación, que ofrece una interfaz bien definida para la funcionalidad que se invoca.[15] |

Tabla 3 Glosario

# Contexto del proyecto

En esta se detallarán los modelos de proceso contemplados por el equipo de trabajo, individualmente detallados y cuál se usará como guía del proyecto, únicamente se encontrarán metodologías agiles, ya que para el grupo resultan de mayor valor para el desarrollo de la idea, esto por sus filosofías, modelo de estructuración y versatilidad además de transparencia.

## Modelo de ciclo de vida

### Metodologías agiles: *Scrum*

Scrum es un marco de trabajo, mediante el cual las personas pueden abordar problemas complejos a la vez que entregan productos del máximo valor posible [16], mediante una metodología productiva e innovadora.

La metodología *Scrum* se fundamenta en tres elementos base: roles, procesos y artefactos.

Existen tres roles en *Scrum*, estos son:

* *Product* *Owner*:

Es el responsable de establecer contacto con los *stakeholders*, transformar las ideas en requisitos y maximizar el valor de producto resultante del trabajo del equipo de desarrollo.[16]

* *Scrum* *master*:

Es un guía en el equipo *Scrum* el cual sirve tanto al equipo de desarrollo como al *product* *owner*, entre sus objetivos está el mantener los procesos adecuados de *Scrum* y ayudar a entender qué interacciones del personal externo pueden ser útiles o no para el *Scrum* *team*. [16]

* *Scrum* *team*:

Es un equipo autogestionable y autocoordinado el cual está en la capacidad de desarrollar los requisitos entregados por el *product* *owner*, son los únicos capaces de realizar incrementos en el proyecto para poder al final de cada *Sprint* entregar una parte del proyecto a producción.

Los pilares fundamentales de la metodología son la transparencia, la inspección y la adaptación, además de ello este marco de trabajo se basa en 5 valores fundamentales:

* Compromiso
* Coraje
* Foco
* Apertura
* Respeto

Entre los artefactos se encuentran:

* *Product* *Backlog*: Es una pila en la cual en la cima se encuentran los requisitos más importantes, estos pueden contener historias de usuario y puede ser extendida con el tiempo tomando en cuenta los incrementos.
* *Sprint* *Backlog*:

Es una pila específica para un determinado *Sprint*, en el cual los desarrolladores predicen los elementos que serán implementados en esa vuelta, dichas características serán implementadas en el incremento final o terminado.

* Incremento: El Incremento es la suma de todos los elementos de la Lista de Producto completados durante un *Sprint* y el valor de los incrementos de todos los *Sprint’s* anteriores. [16]

### *Extreme* *programming*

Es un marco de desarrollo ágil, su objetivo es entregar *software* de alta calidad y una mejor calidad de vida para el equipo de desarrollo, su diferencia con *Scrum* recae en el nivel de especificidad de requisitos.

Desarrollada para abordar las necesidades específicas del desarrollo de software mediante pequeños equipos, esto frente a objetivos cambiantes.

Se fundamenta en valores y principios específicos, además, el costo de cambiar o modificar un programa puede mantenerse constante a lo largo del tiempo.

Entre sus características se encuentran:

* Pruebas unitarias
* Diseños simples y rediseños
* Retroalimentación
* Comunicación abierta y fluida

Las prácticas de *XP* han cambiado un poco desde que se introdujeron inicialmente. Las doce prácticas originales se enumeran a continuación[17]:

* El juego de planificación
* Pequeñas versiones
* Metáfora
* Diseño simple
* Pruebas
* Refactorización
* Programación de pares
* Propiedad colectiva
* Integración continua
* Semana de 40 horas
* Cliente en el sitio
* Estándar de codificación

### Kanban

Es un sistema de flujo que permite mediante el uso de señales la movilización de unidades a través de una línea de producción mediante una estrategia de jalonamiento[18].

El modelo tradicional de Kanban consta de un grupo de tarjetas las cuales representan requisitos o cosas por hacer, las cuales circulan a través de circuitos llamados comúnmente pendientes, en revisión, en curso, bloqueadas y lista. [19]

Entre los múltiples beneficios de Kanban se encuentran el poder detectar algún incidente en determinada fase del proyecto, al ser tan flexible permite reorganizar la carga de trabajo y determinar qué tanto está haciendo un individuo.

La organización y colaboración es fundamental en Kanban gracias a que, al ser un método visual, gestiona adecuadamente múltiples parámetros por medio de colores y guías que ayudan a que el equipo tenga una mayor percepción de donde se está mediante la memoria visual. [20]

Los participantes del proyecto pueden decidir qué tareas trabajar, por tanto, es la persona quien establece su carga de trabajo, así mismo se pueden organizar estructuras jerárquicas basadas en roles prestablecidos identificados con distintivos o colores.

### Metodología seleccionada

La metodología escogida por el grupo es el marco de trabajo *Scrum*, esto ya que nos permite llegar a las necesidades específicas del proyecto, al ser algo complejo y gestionable podemos establecer roles de trabajo y priorizar tareas mediante el *Product* *backlog* ver [Anexo A(Actas)](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Actas), adicionalmente a ello se estableció una modalidad de trabajo semanal con reuniones periódicas identificadas bajo el siguiente esquema:

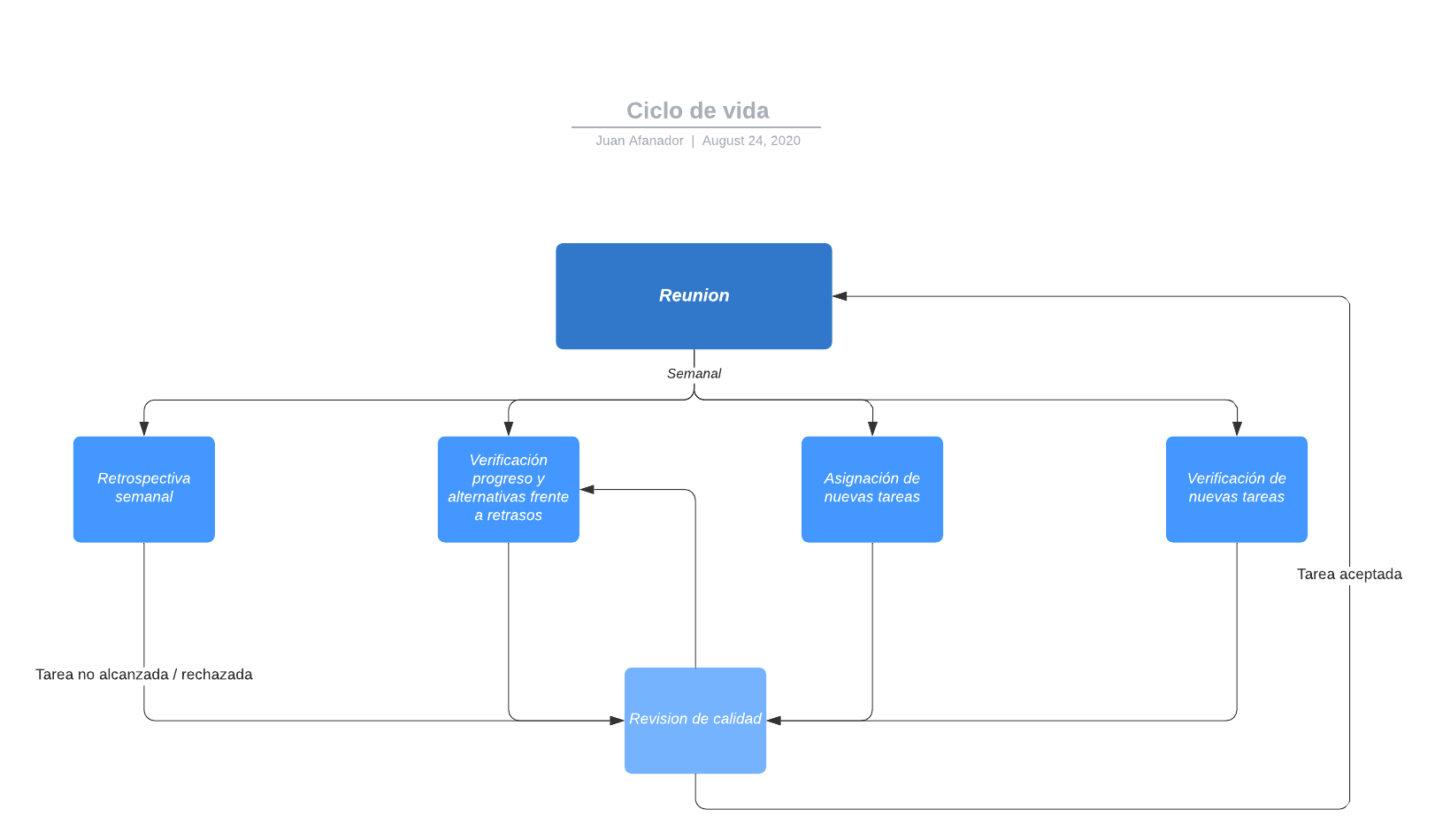


Figura 1 Modelo de ciclo de vida ECO

Imagen 1. Modalidad de trabajo.

Esto nos permite revisar semanalmente los incidentes, aciertos y demás eventos producidos en cada vuelta de incremento con respecto al SPMP, para la gestión se realizó una distribución acorde a un modelo KANBAN, donde las tareas serán administradas por el *Product* *Owner* y el *Scrum* *Master* en un tablero dividido en 4 partes: por hacer, en revisión, realizado, cancelado.

Una vez se realicen las verificaciones pertinentes con respecto a la calidad de la tarea, se evaluará su adición al proyecto y al documento de manera modular, es decir, sin invadir la integridad general del aplicativo, y, al finalizar, se asignarán nuevas tareas para volver al ciclo de vida.

## Lenguajes y herramientas

A continuación, se presentarán los lenguajes y herramientas a utilizar durante el desarrollo del proyecto; las cuales se seleccionaron con los criterios definidos por el grupo

### Herramientas y lenguajes de desarrollo

Para la selección de lenguajes y herramientas se presentan los siguientes criterios:

1. **Curva de aprendizaje:** Los lenguajes y las herramientas seleccionadas deben tener una sintaxis clara, además de contar con soporte y comunidad para una solución eficiente a los problemas que se presenten, y con esto simplificar el trabajo.
2. **Tecnología web:** La tecnología seleccionada permite la ejecución del proyecto en cualquier navegador web, con la posibilidad de agregar el formato especifico de cada navegador para su correcta visualización.
3. **Fácilmente escalable:** La tecnología seleccionada permita trabajar en módulos para agregar nuevas funcionalidades al proyecto, sin que se generen conflicto con las versiones anteriores de este.
4. **Nivel de personalización:** La tecnología cuenta con componentes personalizables, los cuales se pueden adaptar con los dispositivos que se estén utilizando.

### Desarrollo Web

Para el desarrollo se tuvieron en cuenta los entornos de desarrollo, *frameworks*, lenguajes y librerías. De las cuales se encuentran las siguientes alternativas:

* **Visual Studio:** Es un editor de código fuente desarrollado por *Microsoft*, ligero pero potente que se ejecuta en escritorio y está disponible para *Windows*, *macOS* y *Linux*. Viene con soporte incorporado para *JavaScript*, *TypeScript* y *Node.js* y tiene un rico ecosistema de extensiones para otros lenguajes.[21]
* **Atom:** Es un editor de texto de código abierto creado por *GitHub* (ahora propiedad de *Microsoft*) que permite personalizar fácilmente todos sus aspectos. Toda la aplicación está escrita en *JavaScript*, CSS y HTML. Este se encuentra disponible para *Windows*, *macOS* y *Linux*.[22]
* **HTML:** Significa **Lenguaje de Marcado de Hipertextos (*HyperText* *Markup* *Language*)**, es la pieza más básica para la construcción de la web y se usa para definir el sentido y estructura del contenido en una página web.[23]
* **Bootstrap:** Es un *framework* *front*-*end* gratuito para desarrollo web rápido y sencillo, este incluye plantillas de diseño basadas en HTML y CSS para tipografía, formularios, botones, tablas, navegación, etc, así como complementos de *JavaScript* opcionales. [24]
* **Foundation:** Un *framework* para cualquier dispositivo, medio y accesibilidad. *Foundation* es una familia de marcos *front*-*end* receptivos que facilitan el diseño de sitios web, aplicaciones y correos electrónicos receptivos que se ven en cualquier dispositivo. *Foundation* es semántico, legible, flexible y completamente personalizable. [25]
* **React:** Es una librería de *JavaScript* para crear interfaces de usuario. *React* facilita la creación de interfaces de usuario interactivas utilizando vistas declarativas que hacen que el código sea más predecible, más simple de entender y más fácil de depurar. Dado que la lógica de los componentes está escrita en *JavaScript* en lugar de en plantillas, puede pasar fácilmente datos enriquecidos a través de su aplicación y mantener el estado fuera del DOM.[26]
* **Vue:** Es un *framework* progresivo para construir interfaces de usuario. Vue está diseñado desde cero para ser adoptable gradualmente. La biblioteca principal se centra solo en la capa de vista y es fácil de recoger e integrar con otras bibliotecas o proyectos existentes. Por otro lado, Vue también es perfectamente capaz de impulsar aplicaciones sofisticadas de una sola página cuando se usa en combinación con herramientas modernas y bibliotecas de soporte.[27]
* **Angular:** Es un *framework* de diseño de aplicaciones y una plataforma de desarrollo para crear aplicaciones de una sola página eficientes y sofisticadas.

Angular está escrito en *TypeScript*. Angular utiliza componentes los cuales definen vistas, que son conjuntos de elementos de pantalla entre los que Angular puede elegir y modificar de acuerdo con la lógica y los datos de su programa.[28]

Las herramientas y tecnologías seleccionadas para este proyecto son

* **Visual Studio Code:** aunque tanto VSC como Atom poseen gran variedad de *plugins* personalizables, se decidió por escoger VSC como el editor sobre el cual se desarrollará el proyecto, ya que este ofrece mejores *plugins* para el desarrollo web bajo las tecnologías seleccionadas (descritas a continuación). Además, el grupo de trabajo tiene experiencia en el manejo de este editor.
* **HTML, CSS y JavaScript:** estas tecnologías son obligatorias en el desarrollo web, ya que no hay otras opciones en el mercado.
* **React:** se escogió dado esta librería dado las comodidades que proporciona el uso de componentes y módulos para la construcción de sitios web, lo que permite dividir de forma más cómoda el trabajo entre los integrantes del equipo. Adicionalmente, esta librería es una de las más usadas en el mundo del desarrollo, lo que da una ventaja en cuanto a documentación y aprendizaje refiere. Otro criterio para seleccionar esta tecnología es que los integrantes del equipo de trabajo tienen intereses personales en aprender a usarla, dado que ninguno tiene conocimientos previos en su uso.
* **Bootstrap:** se escogió este *framework* ya que provee muchas herramientas para la personalización de componentes gráficos, lo que ahorra tiempo en el desarrollo general de la interfaz gráfica de usuario y provee un diseño más amigable, además disminuye la carga de trabajo con *CSS*. Otra razón por la que se escogió *Bootstrap* es por su facilidad de integración con *React*.

#### **Base de datos y despliegue de la aplicación**

Para seleccionador el manejador de base de datos y el despliegue de esta se tomaron estos criterios:

1. Debe proveer un servicio de *hosting* que permita el fácil despliegue de la aplicación.
2. La base de datos debe ser en tiempo real.
3. La herramienta de despliegue debe proveer mecanismos para la autenticación de usuario.

* **Firebase:** Es una plataforma de desarrollo de aplicaciones web y móviles, respaldada por Google. *Firebase* administra su propia infraestructura con un buen conjunto de herramientas para simplificar el flujo de trabajo del desarrollador proporcionándoles kits de desarrollo y un panel en línea. Estos kits de herramientas están interconectados, son escalables e integrables con *software* de terceros para superar desafíos complejos con bloques de construcción estándar. *Firebase* cuenta con *Realtime* *Database* y *Cloud* *Firestore* que pueden almacenar datos estructurados en documentos y sincronizar las aplicaciones correspondientes en milisegundos cada vez que se produce una transformación de datos.[29]
* **Azure:** Azure *Database* ofrece una selección de bases de datos relacionales, *NoSQL* y en memoria totalmente administradas, que abarcan motores patentados y de código abierto, para adaptarse a las necesidades de los desarrolladores de aplicaciones actuales. La administración de la infraestructura, incluidas la escalabilidad, la disponibilidad y la seguridad, está automatizada, lo que ahorra tiempo y dinero. Azure proporciona información sobre el rendimiento con el uso de inteligencia integrada, modifica su escala sin límites y administra las amenazas de seguridad.[30]

Se ha tomado la decisión de implementar como base de datos *Firebase*, ya que cumple con los requisitos mencionados anteriormente, además de que hay integrantes que ya han manejado esta tecnología.

### Herramientas de control de versiones y hosting de versiones

Para las herramientas de control de versiones y *hosting* se definieron los siguientes criterios:

1. La herramienta debe proveer de manera completa todas las opciones para un correcto manejo de las versiones de los ítems de configuración durante el proyecto.
2. El equipo debe tener estar familiarizado con la herramienta elegida.
3. Debe tener una curva de aprendizaje moderada.

Se consideraron dos herramientas para el manejo del control de versiones en el proyecto:

* **Git:** Es el sistema de control de versiones más utilizado en la actualidad. Git es un sistema de control de versiones distribuido, lo que significa que su copia local del código es un repositorio de control de versiones completo. Estos repositorios locales completamente funcionales hacen que sea fácil trabajar sin conexión o de forma remota. Confirma su trabajo localmente y luego sincroniza su copia del repositorio con la copia en el servidor. La flexibilidad y la popularidad de Git lo convierten en una excelente opción para cualquier equipo. [31]
* **Subversión:** Es una herramienta que se utiliza para mantener versiones actuales e históricas de proyectos. Subversión es un sistema de control de versiones centralizado de código abierto. Tiene licencia de Apache. [32]

Ya planteadas estas dos herramientas de control de versiones, se ha tomado la decisión de trabajar con Git, ya que cumple con los criterios, además de ser una de las herramientas más utilizadas en el mercado laboral.

Para la plataforma de hosting de versiones, se consideraron las siguientes:

* **GitHub:** Es una plataforma de alojamiento de código para el control de versiones y la colaboración. Permite a las personas trabajar juntos en proyectos desde cualquier lugar.[33]
* **GitLab:** Es una plataforma *DevOps* completa, entregada como una sola aplicación. Esto hace que GitLab sea único y crea un flujo de trabajo de software optimizado, liberando a su organización de las limitaciones de una cadena de herramientas ensamblada. GitLab ofrece una visibilidad incomparable y niveles más altos de eficiencia en una sola aplicación a lo largo del ciclo de vida de DevOps. [34]

Se ha elegido GitHub ya que los integrantes han tenido experiencia en esta herramienta. Además, con la explicación dada en clase queda claro las posibilidades que se tiene con le elección de esta aplicación.

### Herramientas de Modelado

Para la selección de herramientas de modelado, se definieron los siguientes criterios:

1. Debe proveer la gran mayoría de diagramas *UML*.
2. Debe proveer colaboración para equipos.
3. Los miembros del equipo deben estar familiarizados con la plataforma.

Las herramientas consideradas para el modelado *UML* fueron las siguientes:

* **StarUML:** Es una plataforma de modelado de software que soporta *UML* (*Unified* *Modeling* *Language*). Se basa en la versión 1.4 de *UML* y proporciona once tipos diferentes de diagramas, y acepta la notación *UML* 2.0. Sobresale por su adaptabilidad al entorno del usuario y tiene una alta extensibilidad en su funcionalidad. Garantiza la maximización de la productividad y la calidad de sus proyectos de *software*. [35]

Se ha tomado la decisión del uso de *StarUML* ya cumple los criterios planteados. Además de que la pagina provee gran número de diagramas *UML* y permite la colaboración entre los miembros del trabajo en equipo.

#### **Modelado de procesos de negocio**

En cuanto al modelamiento BPMN las herramientas consideradas fueron:

* **Visual Paradigm:** Es una herramienta de software diseñada para que los equipos de desarrollo de software modelen el sistema de información empresarial y gestionen los procesos de desarrollo. Esta herramienta soporta los principales lenguajes y estándares de modelado de la industria. Ofrece un completo conjunto de herramientas de software que las empresas necesitan para la captura de requerimientos, análisis de procesos, diseño de sistemas, diseño de bases de datos, etc. [36]
* **Bizagi Modeler:** Es una aplicación gratuita para diagramar, documentar y simular gráficamente procesos para negocios que se rigen por las directrices del entrono PBMN 2.0. [37]

Mirando estas alternativas para la realización de los diagramas BPMN se ha tomado la decisión de utilizar Bizagi Modeler, ya que cumple los requisitos. Por otra parte, todos los integrantes del grupo ya lo han utilizado y tienen experiencia en esta herramienta.

### Manejo de documentos

Se definieron los siguientes criterios para la selección de las herramientas para el manejo de documentos:

1. Debe tener alojamiento en la nube, que permita la colaboración en tiempo real del equipo de trabajo.
2. Debe proveer una variedad de herramientas ofimáticas.
3. Debe proveer herramientas para el versionado de los documentos.

Las herramientas consideradas fueron las siguientes:

* **G Suite:** Es un conjunto de aplicaciones web creadas por *Google*. A través de esta herramienta tendrá acceso a una serie de potentes aplicaciones de *Google* la cual tendrá la capacidad de compartir hojas de cálculo y documentos, crear videoconferencias con *Hangouts* y utilizar la mensajería instantánea. También puede compartir calendarios con otras personas, lo que facilita la planificación de reuniones. [38]
* **Microsoft Office 365:** Es una versión basada en la web del conjunto de aplicaciones de productividad empresarial de *Microsoft* *Office*. *Office* 365 se entrega a los usuarios a través de la nube e incluye *Exchange* *Online* para correo electrónico, *SharePoint* *Online* para colaboración, *Lync* *Online* para comunicaciones unificadas y un conjunto de Office Web Apps, versiones basadas en Web del conjunto de aplicaciones tradicional de *Microsoft* *Office*.[39]

Se ha escogido la herramienta *Microsoft* *Office* *365*, ya que contamos con la licencia dada por la universidad, además de poder trabajar en tiempo real en sus herramientas ofimáticas, visualizando el trabajo realizado por cada integrante, además esta herramienta nos ofrece la comunicación entre el grupo por medio de *Microsoft* *Teams* y dejar registro de nuestras reuniones.

### Manejo del proyecto

Para la selección de una herramienta para el manejo del proyecto, se especificaron los siguientes criterios:

1. Debe proveer una forma clara y sencilla para el manejo de proyectos.
2. Debe proveer facilidad a la hora de asignar tareas a los miembros del equipo.
3. Los miembros del equipo deben tener experiencia con la herramienta seleccionada.
4. Debe proveer notificaciones sobre las tareas asignadas en tiempo real.

Una vez planteados estos criterios se consideraron las siguientes alternativas:

* **Trello:** Es una herramienta de colaboración que organiza proyectos en tableros. A través de Trello podrá saber cuáles son las tareas que se llevan a cabo, quién trabaja en una tarea determinada y cuál es el estado de un proceso.[40]
* **Monday:** Es un sistema operativo de trabajo (Work OS) que permite a los equipos ejecutar proyectos y flujos de trabajo con confianza. Es un simple, pero intuitivo, Sistema Operativo de Trabajo para que los equipos den forma a los flujos de trabajo, se ajusten a las necesidades cambiantes, creen transparencia y se conecten en colaboración. Este es personalizable y puede ser usado para cualquier caso de uso.[41]

Se ha tomado la decisión de trabajar con ambas herramientas ya que Trello nos ofrece *power*-*ups* que facilitan la integración con otras herramientas. Esto es una gran ventaja a la hora de centralizar todas las tareas y procesos en materia de administración y gestión del proyecto. Para poder realizar el seguimiento de las tareas en un diagrama de Gantt se ha utilizado *Monday* ya que es gratuita y permite llevar un seguimiento a cada actividad con una gran personalización.

## Plan de aceptación del producto

En el siguiente numeral se establecen los diversos requisitos y criterios establecidos con el *stakeholder* para poder llevar a ejecución el proyecto, de igual manera se detallan herramientas y técnicas empleadas para una correcta dirección de este.

### Entregables

En la siguiente tabla se detalla cada uno de los documentos a presentar a lo largo del desarrollo del proyecto:

|  |  |  |
| --- | --- | --- |
| Entregable | Criterios | Técnicas y Herramientas |
| SPMP | Máximo 50 páginas, referencias IEEE, sin errores de ortografía, redacción clara y concisa, enumeración correcta de tablas y figuras y tabla de contenido bien definida. | Microsoft Word, para la numeración, tabla de contenido, ortografía, tablas y tablas.  Mendeley, para el manejo de las referencias bibliográficas. |
| SRS | Máximo 30 páginas, referencias IEEE, Sin errores de ortografía, redacción clara y concisa, enumeración correcta de tablas y figuras y tabla de contenido bien definida. | Microsoft Word, para la numeración, tabla de contenido, ortografía, tablas y tablas.  Mendeley, para el manejo de las referencias bibliográficas. |
| SDD | Máximo 25 páginas, referencias IEEE, sin errores de ortografía, redacción clara y concisa, enumeración correcta de tablas y figuras y tabla de contenido bien definida. | Microsoft Word, para la numeración, tabla de contenido, ortografía, tablas y tablas.  Mendeley, para el manejo de las referencias bibliográficas. |
| Primer prototipo | Debe estar debidamente documentado, implementar el caso de uso más complejo, código ordenado, debe usar patrones de diseño, debidamente probado. | Pruebas unitarias para detección de errores, uso de versionamiento para llevar un seguimiento al proceso de desarrollo. |
| Segundo prototipo | Debe estar debidamente documentado, implementar al menos el 70% de los casos de uso, código ordenado, debe usar patrones de diseño, debidamente probado. | Pruebas para detección de errores, uso de versionamiento para llevar un seguimiento al proceso de desarrollo. |

Tabla 4 Entregables con criterios y herramientas

## Organización del proyecto y comunicación

Se enuncian los *stakeholders* involucrados en el proyecto, así como la descripción y el organigrama del equipo de trabajo, los roles detallados y las responsabilidades destinadas para cada uno de ellos.

### Interfaces externas o *stakeholder*

En la siguiente tabla se evidencian los *stakeholders*, personas interesadas y clientes del proyecto, al igual que las interfaces externas.

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre Entidad o *Stakeholder* | Descripción | Responsabilidad | Datos de contacto |
| Carlos Andrés Parra | *Stakeholder*: Profesor de catedra de la Pontificia Universidad Javeriana. Director de la asignatura de Ingeniería de *Software* | -Encargado de proveer los lineamientos respecto a cada entrega del proyecto.  -Revisar y aprobar las entregas parciales del proyecto.  - Guiar el proyecto de *software*. | Correo:  ca.parra@javeriana.edu.co |
| *Firebase* | Proveedor del servidor web y base de datos a usar. | -Responsable de la implementación del *Backend* del proyecto. Apoyando la interacción entre el cliente, el servidor y su base de datos. | Pagina web: <https://firebase.google.com/?hl=es-419> |
| |  | | --- | | *Grupo ECO* | | Grupo de estudiantes de la Pontificia Universidad Javeriana | -Responsables de entregar un proyecto que cumpla con los objetivos y necesidades planteadas  -Cumplir con la fecha de la entrega final y las parciales del producto | Correo:  buenvecinoeco@gmail.com |
| *Arrendatarios/*  *Clientes* | -Usuarios principales del proyecto | -En caso de ser cliente, hacer buen uso del espacio donde esta residiendo, cumplir con la normativa del lugar y cumplir con el pago en la fecha acordada  -En caso de arrendatario, prestar un inmueble, en buen estado y que cumpla con los requisitos descritos previamente | -La información del contacto se recibe cuando el usuario se regristra |

Tabla 5 Skateholders

### Organigrama y descripción de roles

Para que el grupo tuviera una distribución de carga justa y eficiente se llevó a cabo el proyecto por medio de la metodología scrum, esto quiere decir que para cada miembro se estableció un rol especifico (*Scrum* *master*, *Product* *Owner*, *Development* *team*) de acuerdo a sus habilidades y gustos, esto contrastado con pruebas de personalidad ver anexo ([Test Hermann](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Test%20Herrmann))

#### **Organigrama**

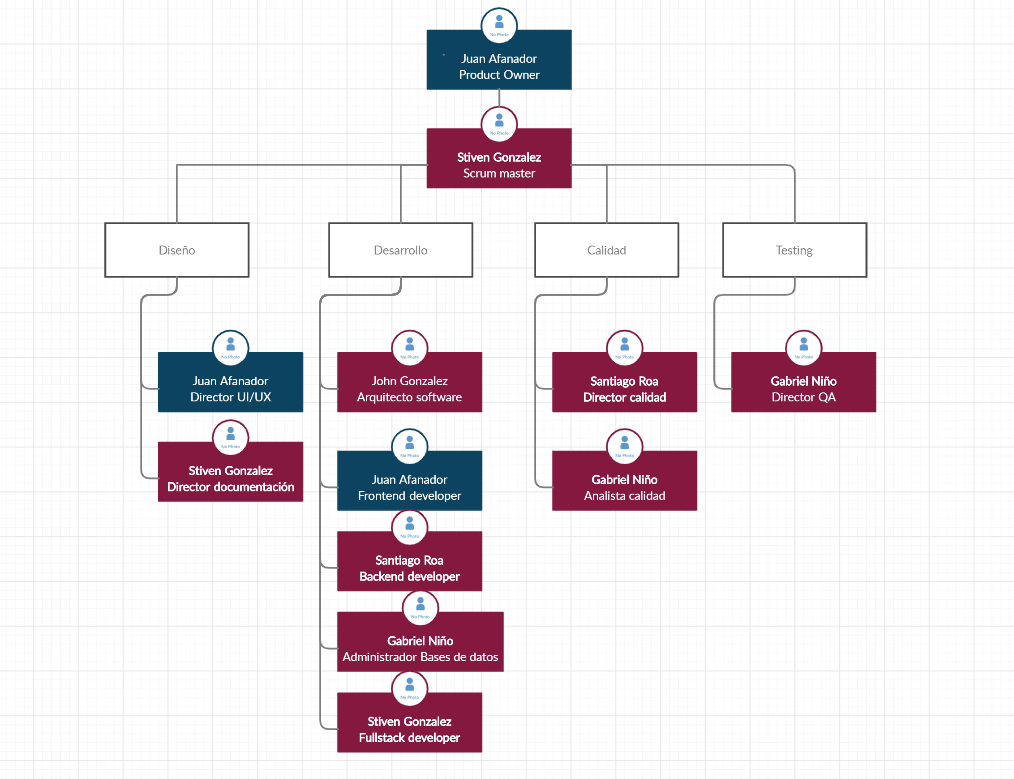


Figura 2 Organigrama equipo de trabajo ECO

|  |  |  |
| --- | --- | --- |
| **Rol** | **Descripción** | **Responsabilidades** |
| *Product* *owner* | Se encarga de coordinar principalmente la gestión de requisitos. Es el que tiene la última palabra en las decisiones del proyecto | * Revisar el progreso de las funcionalidades en cada requisito. * Verificar el desempeño de todos los miembros del equipo. * Diseñar e implementar los índices de calidad. * Encargado de revisar las recomendaciones planteadas por el docente. * Diseñar planes para el manejo y control de riesgos. |
| *Scrum* *master* | Responsable de asegurar que *Scrum* es entendido y adoptado a lo largo del proyecto. También se debe mantener al equipo unido, solucionando los problemas que puedan ocurrir entre los miembros del equipo a lo largo del proyecto. | * Hacer cumplir el cronograma. * Proteger al equipo de interrupciones y distracciones. * Resolver conflictos internos. * Hacer respetar las decisiones tomadas democráticamente. * Asegurar la implementación de *Scrum*. * Organizar reuniones del equipo. |
| Director UI/UX | Se encarga de desarrollar las interfaces de la aplicación y la experiencia del usuario con la misma. | * Recopilar y evaluar los requisitos de los usuarios, en colaboración con el *Scrum* *Master* y *Project* *Manager*. * Ilustración de ideas de diseño utilizando guiones gráficos, flujos de procesos y mapas de sitio. * Diseñar elementos gráficos de la interfaz de usuario, como menús, pestañas y *widgets*. |
| Director documentación | Responsables de asegurar y monitorear la documentación del proyecto en todas sus etapas, tanto en el código de fuente como en el diseño del sistema. | * Definir el tipo de documentación que se necesita (técnica, manuales). * Monitorear la documentación tanto del código como del diseño. * Informar sobre la documentación al director de proyecto. * - Asociarse con los directores de desarrollo y configuración para generar los manuales de instalación |
| Arquitecto *software* | Encargados de planear la arquitectura tanto lógica como física del proyecto de acuerdo con los requisitos planteados. | * Elegir las tecnologías para el proyecto a lo largo del semestre. * Desarrollar las funcionalidades con buenos estándares de programación, en lo posible el uso de patrones de diseño. * Limitarse a la arquitectura diseñada desde un principio. * Verificar que todos los componentes estén bien acoplados. * Comunicar cualquier novedad al *Scrum* *Máster*. * Levantar requerimientos. * Plantear la arquitectura del sistema. |
| *Frontend* *developer* | Desarrollador enfocado en la parte principal del sistema, enfocado en mejoras visuales y la interacción del usuario con el sistema. | * Traducir los requisitos e ideas en maquetación responsiva * Verificar el correcto funcionamiento de componentes visuales * Lograr una correcta adecuación y comunicación entre los componentes lógicos y funcionales con los componentes gráficos |
| *Backend* *developer* | Desarrollador enfocado en el funcionamiento trasero de la aplicación, su función es crear funcionalidades a nivel funcional sin importar la calidad grafica del proyecto. | * Generar módulos acoplables y funcionales que pueden ser incorporados por los desarrolladores *front* *end*. * Verificar correcta funcionalidad de bases de datos y su correcta integración * Conexión de parte frontal con funcionalidades no visibles para clientes. |
| Administrador bases de datos | Encargado de la gestión de la base de datos a utilizar durante el proyecto. | * Administrar la estructura de la Base de Datos. * Administrar la actividad de los datos. * Administrar el Sistema Manejador de Base de Datos. * Establecer el Diccionario de Datos. * Asegurar la confiabilidad de la Base de Datos. * Confirmar la seguridad de la Base de Datos. |
| Director calidad | Se encargan de coordinar y monitorear las actividades relacionadas con gestión y control de la calidad en todas sus áreas. Así como los posibles riesgos para el proyecto y de gestionar planes para el tratamiento de éstos. | * Revisar ortografía, forma, coherencia y cohesión del trabajo escrito. * Verificar el desempeño de todos los miembros del equipo. * Diseñar e implementar los índices de calidad. * Encargados de revisar las recomendaciones planteadas por el docente. * Diseñar planes para el manejo y control de riesgos. |
| Analista calidad | Miembro de apoyo y soporte para gestión de procesos de calidad y sostenibilidad del aplicativo | * Verificar mediante una segunda opinión la integridad y funcionamiento del aplicativo frente a diversos problemas, esto de la mano del director de QA |
| Director QA | Se asegura de que la aplicación cumpla adecuadamente la administración del cambio, la gestión de versiones, construcción del sistema y gestión de entregas. | * Guiará a los demás integrantes del grupo en el manejo del sistema de gestión de versiones. * Creará los repositorios necesarios para gestionar el almacenamiento de la información. * Encargado de llevar al tanto del directorio de versiones de los ítems de configuración. |

Tabla 6 Descripción de roles

# Administración del proyecto

## Métodos y herramientas de Estimación

Un proyecto de *software* está compuesto de múltiples piezas, y se conocen varias técnicas de estimación; como lo son las empíricas o las que usan modelos algorítmicos de costos, en los cuales se usa una fórmula matemática que sirve para predecir los costos del proyecto, en los que se tienen en cuenta el tamaño del *software*. [42]

A pesar de las estimaciones mencionadas anteriormente, es necesario aplicar técnicas de descomposición, ya sea a un problema o proceso, para que de esta forma el proyecto sea más fácil de organizar. La estimación que se basa en el proceso se concentra en estimar el esfuerzo para terminar cada uno de estos [43]. Sin embargo, la estimación basada en el problema busca dimensionar cada elemento de *software* por medio de métricas de productividad las cuales son las líneas de código y los puntos de función. [43]

Las estimaciones mencionadas anteriormente son las clásicas, sin embargo, como se mencionó previamente, para poder realizar el proyecto se va a usar la metodología *Scrum*, por ende, técnicas de estimaciones agiles. Así que, se va a implementar la técnica *Planning Poker* la cual se explicará a continuación.

### *Planning* *Poker*

Se escogió este método principalmente porque permite conocer la opinión de cada miembro de grupo y le concede al grupo tener un control acerca de las actividades que se han realizado a lo largo de la semana.

Para poder empezar con él proceso se tuvo que observar el tiempo disponible para finalizar el proyecto, ya que con este es que se iba a platear el tiempo límite, con el objetivo de poder organizar todas las historias de usuario planteadas en el *Post-it*,([Ver anexo Miro](https://miro.com/app/board/o9J_kncNyAM=/)), por medio del cual se va a actualizar cada que se cumpla con una de estas historias. Se concluyó, que se debían realizar estas historias en un intervalo de 10 semanas, es decir, 70 días.

Para poder realizar la actividad de forma virtual, se uso <https://www.planningpoker.com/> la cual nos permite votar por cada historia de usuario con el valor que uno cree que es el pertinente. La persona encargada de la actividad, que en este caso fue el *Product Owner*, Juan José Afanador, era el encargado de revisar los valores obtenidos al final de la votación, en caso de tener una diferente valoración numérica se planteaba una discusión en la cual los miembros del grupo que tuviera el valor más alto y el más bajo decía las razones de ese resultado, se realizó este proceso hasta que todas las historias de usuario de cada sprint tuvieran su valoración.

Esta valoración representa el número de días ideales en los que se debe completar una historia de usuario, la valoración que se obtuvo para cada sprint fue de 10 días en el primer sprint, 19 días en el segundo y y 20 días en los últimos dos sprint, esto se realizará inicialmente con el objetivo de mejorar el control que se tenga del trabajo realizado en el tiempo estipulado, sin embargo, los dos ultimos sprints van a compartir una historia de usuario para poder completar los 20 días en cada una de ellas, eso para que se puedan realizar las pruebas pertinentes a cada uno y de esta forma garantizar el correcto funcionamiento de cada uno. Como se mencionó al principio de esta sección se cuentan con 70 días, en los cuales se van a trabajar arduamente para poder cumplir con lo estipulado en el *Planning Poker*.

A continuación, se mostrará el puntaje que tuvo cada historia de usuario en los sprints:

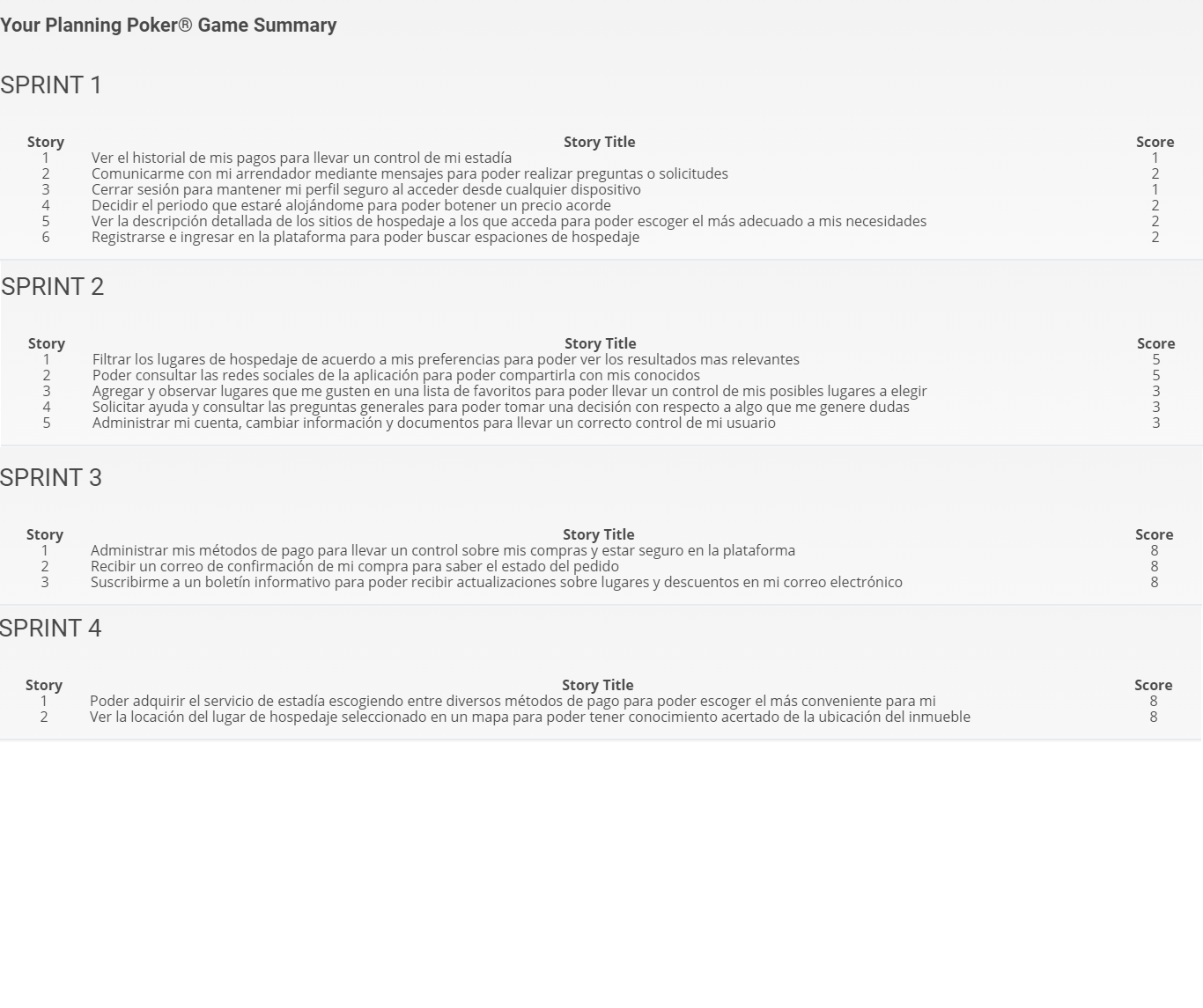


Figura 3 Planning Poker Sprint

## Inicio del proyecto

Para el proyecto se tomó en consideración las aptitudes y habilidades de cada miembro según la prueba de Herrmann [ver anexo Herrmann](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Test%20Herrmann), en consecuencia, se identificaron estilos de aprendizaje los cuales pueden ser apostados por planes de capacitación personalizados, esto con el fin de adquirir o asegurar conocimientos para ser luego puestos en práctica.

### Plan de capacitación

El plan de capacitación está orientado a ser totalmente autodidacta previa aprobación del grupo, esto quiere decir que para cada rol se identificaron herramientas de aprendizaje y recursos que facilitan una correcta adaptación del tema.

* ***Scrum* *Master* (Stiven Gonzalez):**

Stiven será el encargado de verificar y gestionar el correcto funcionamiento de *scrum*, esto mediante su rol permanente y la vigilancia sin influir en las decisiones del *development* *team*, su objetivo es asegurarse de la integridad del grupo y es por ello que debe instruirse muy bien en la metodología *scrum*, para ello tendrá que tomar el curso *Applied Scrum for Project Management*[44]*,*el cual ayudará a fortalecer sus habilidades de liderazgo mientras adquiere habilidades blandas.

Su tiempo establecido para la completitud del curso es de 1 semana.

* ***Product* *Owner* (Juan Afanador):**

Encargado de asumir y liderar el grupo al igual que contar con las habilidades blandas y conocimientos para lograr comunicación asertiva y concreta, esto con el objetivo de transmitir correctamente la información del *stakeholder* al *development team* para poder generar un producto a la medida del cliente, para ello este miembro del grupo deberá cursar y aprobar el curso *Scrum* *Foundations* *Professional* *Certificate* (SFPC)[45], esto con el fin de conocer el correcto flujo de scrum y contar con las bases necesarias para liderar un proyecto ágil. Dicha carga de aprendizaje debe ser tomada en módulos semanales de 2 a 3 horas hasta obtener la certificación y pasar el examen.

Adicionalmente a ello será el encargado de la interfaz, Juan es el miembro del grupo el cual ha trabajado con diseño y conoce de herramientas como Figma o Adobe XD, por lo tanto, será el encargado de transcribir las ideas en diseños.

* ***Scrum* *Team* (John González):**

John será el encargado principal de desarrollo, para ello tendrá que capacitarse en la librería *react* por medio del curso *React* *JS* Desde Cero! *Hooks*, *Redux*, *Context*, *Firebase* y más![46], adicionalmente será el encargado de gestionar la calidad del código junto con los demás integrantes del proyecto.

* ***Scrum* *Team* (Santiago Roa):**

Santiago será el encargado principal de la calidad del proyecto, esto quiere decir que llevará una monitorización consecuente con cada documento y versión del código para poder generar módulos adaptables y seguros en el proyecto, adicionalmente será el encargado de crear los diagramas en *UML* y BPMN para tener una mayor claridad del flujo del trabajo, y al igual que John debe contar con conocimiento de *React*.

* **Scrum Team (Gabriel Niño):**

Gabriel es el encargado de las pruebas y bases de datos del proyecto, por lo cual debe conocer muy bien *Firebase* [29]y métodos de *testing* que garanticen una correcta comunicación de los módulos de *React*, para ello deberá estudiar la documentación oficial de *Firebase* [29]y GitHub [33]

### Infraestructura del proyecto

Para poder dar inicio al desarrollo del proyecto es necesario detallar tanto las herramientas físicas como lógicas, con ello se refiere a las características de los equipos de cómputo de cada estudiante al igual que el software a utilizar, por lo tanto,en la tabla 7 se detallarán las herramientas tecnológicas para desarrollar el documento y en la tabla 8 se podrán ver las características técnicas de la infraestructura.

|  |  |
| --- | --- |
| **Nombre herramienta** | **Descripción** |
| *Microsoft* *Word* | Es un procesador de textos publicado por Microsoft que permite crear documentos, informes, cartas y currículos de calidad profesional. Este tiene características como la revisión de la ortografía, la revisión gramatical, el formato de texto y de fuentes y más.[47] |
| *Mendeley* *Desktop* | Es un gestor de referencias gratuito que puede ayudarle a recopilar referencias, organizar sus citas y crear bibliografías.[48] |
| *Creately* | Es una herramienta de diagramación basada en la web, que soporta muchos tipos de diagramas, incluyendo diagramas de flujo, diagramas de Venn, story boards.[48]m |
| *Monday* | Es un simple Sistema Operativo de Trabajo para que los equipos den forma a los flujos de trabajo, se ajusten a las necesidades cambiantes, creen transparencia y se conecten en colaboración. [41] |
| *Trello* | Es una aplicación para la gestión y administración de proyectos, donde se trabaja mediante tableros y varias personas pueden editar y trabajar en ellos.[40] |
| *Planning* *Poker* | Es una técnica de estimación basada en el consenso y el juego, que se utiliza principalmente para estimar el esfuerzo o el tamaño relativo de los objetivos de desarrollo en el desarrollo de software[49] |
| *Bizagi* *Modeler* | Es una herramienta para el modelado de los procesos de negocio.[37] |
| GitHub | Es una plataforma de desarrollo colaborativo, que ofrece a los desarrolladores repositorios de software usando el sistema de control de versiones Git.[33] |
| Canva | Es una herramienta web para crear diseño y gráficos, tales como: logos, presentaciones, tarjetas, entre otros.[50] |
| *StarUML* | Es una herramienta que permite crear diagramas *UML*, como lo son los diagramas de clases, casos de uso, diagramas de actividad, de secuencia, entre otros[35] |

Tabla 7 Herramientas de Software

|  |  |
| --- | --- |
| **Dueño del equipo** | **Características técnicas** |
| Juan Afanador | * Procesador AMD Ryzen 2200g * Memoria RAM de 8GB * Sistema Operativo Windows 10. |
| Stiven González | * Lenovo * Procesador Intel Core i5 * Memoria RAM de 8GB * Sistema Operativo Windows 10. |
| John González | * Lenovo * Procesador Celeron J1800 * Memoria RAM de 4GB * Sistema Operativo Windows 8. |
| Santiago Roa | * Asus * Procesador Intel Core i5 * Memoria RAM de 12GB * Sistema Operativo Windows 10. |
| Gabriel Niño | * Lenovo * Procesador AMD Ryzen 5 3500U * Memoria RAM de 8GB * Sistema Operativo Windows 10. |

Tabla 8 Infraestructura de Hardware

### Tareas para realizar

Para gestionar las tareas a realizar se usó Trello como se muestra en Figura 5, esto ya que al ser una herramienta de gestión grupal permite asignar tareas a cada miembro del grupo con una correcta clasificación según el Sprint y el tiempo en desarrollarse, además de ello se pudo integrar con Monday.com y con Calendar para lograr una mayor perspectiva de a donde se quería llegar. Esto puede ser medido gracias al historial de Trello.

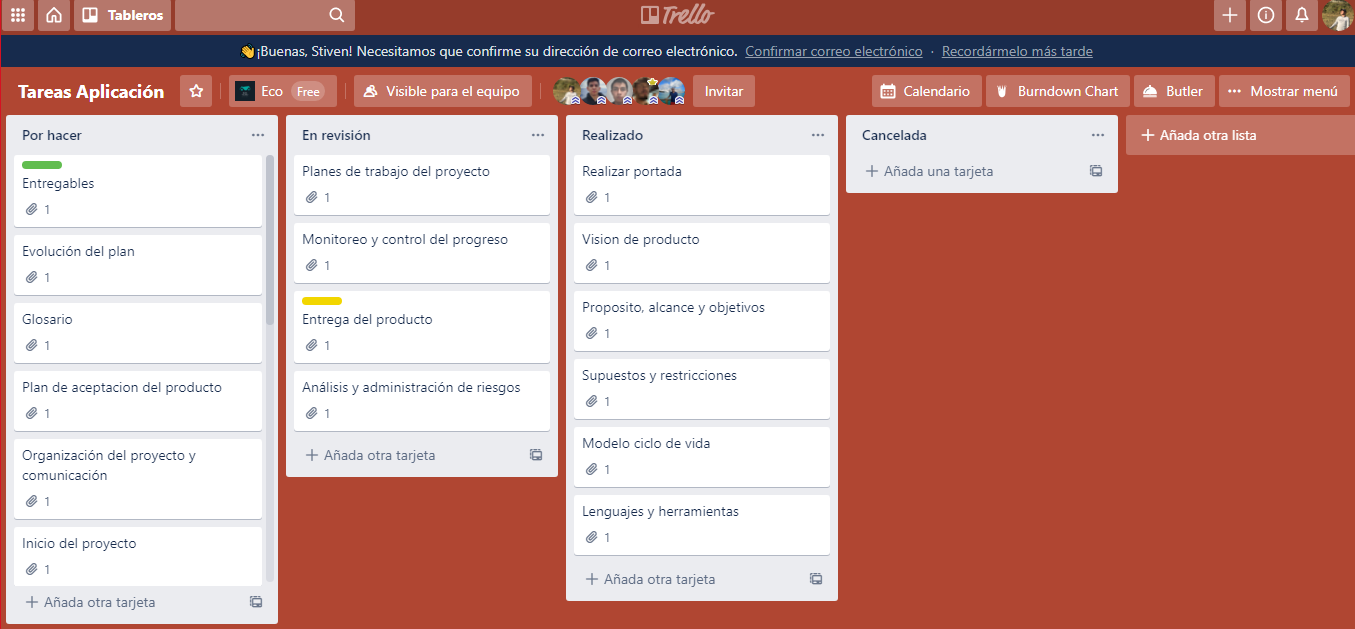


Figura 4 Organización de tareas por Trello

## Planes de trabajo del proyecto

Para la realización del proyecto se consideró importante establecer una ruta de trabajo en la cual se pudiera medir en cuanto al tiempo el avance del proyecto y su costo, esta consta de un diagrama de descomposición de tareas (WBS) al igual que un diagrama de Gantt, esto para conocer las relaciones y dependencias entre tareas a través del tiempo, adicional a ello se construyó un análisis de costos mediante un flujo de cada, en el cual se detalla el costo horario por rol y por Sprint para cada miembro y su impacto monetario total, estas tres componentes juntas permitieron simular el comportamiento a nivel económico de llevar el proyecto a ejecución.

### Estructura de descomposición de tareas (WBS)

Según se refiere [51], WBS se utiliza para representar las tareas por realizar en el proyecto, además de ello establece los límites del mismo. Para la entrega número uno únicamente se contempló como elementos relevantes los ítems más significativos del documento SPMP, y basado en ello se realizó la respectiva grafica. Como se muestra en la figura 5.

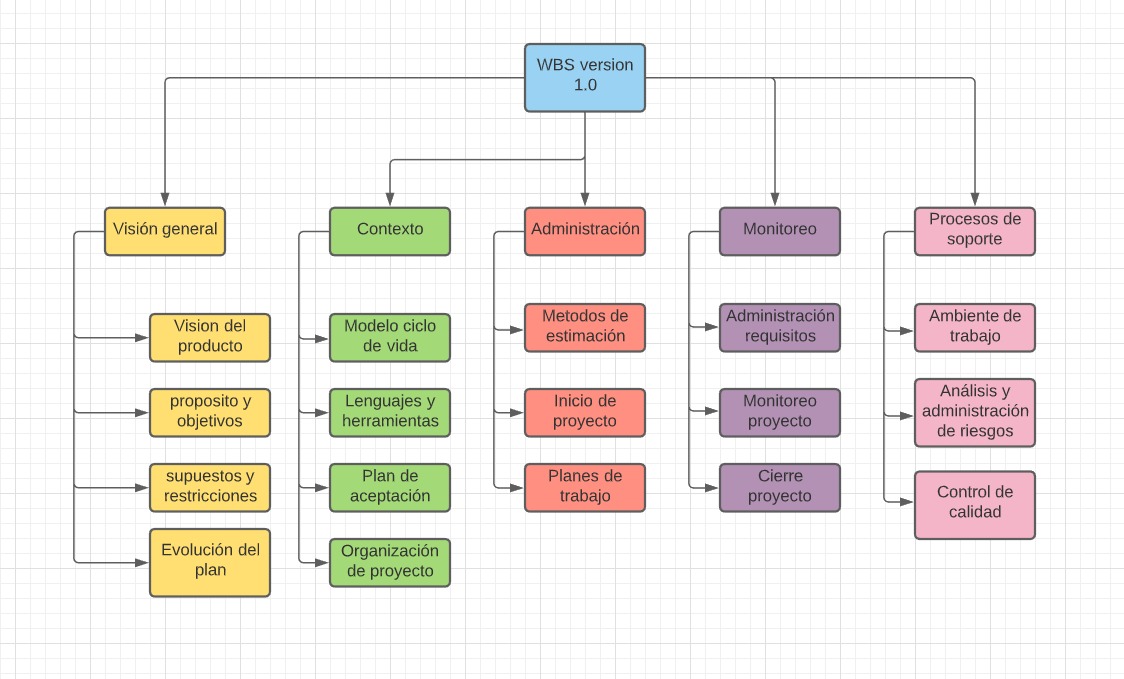
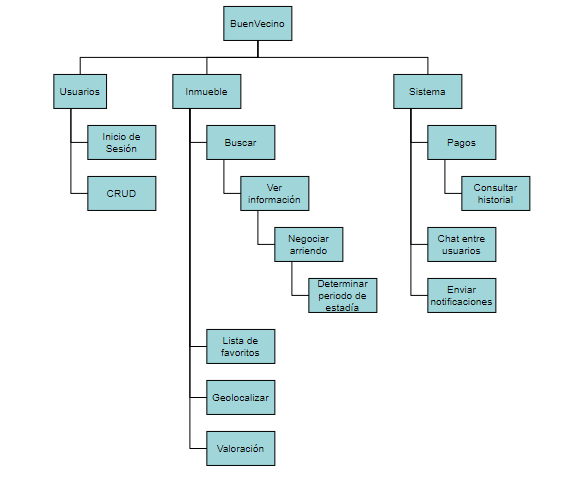


Figura 5 Diagrama WBS



### Calendarización del proyecto

El diagrama de Gantt es una herramienta que facilita la visualización de tareas en una línea de tiempo visual, esto permite al grupo tener una mejor compresión de los objetivos a lograr en periodos semanales y con ello a tener una adecuada gestión de los recursos, todo ello mediante la identificación de puntos críticos en cuanto a fechas, dichos puntos permiten tomar decisiones en cuanto a cómo modificar ítems del proyecto mediante planes de acción que no sean invasivos sino que contemplen un cambio amigable para el proyecto en general.

Para la entrega número uno del proyecto se usó la herramienta *Trello* junto a *Monday* para identificar las fechas de inicio y finalización de cada módulo del documento, esto con sus respectivas necesidades y requerimientos previos y distribución por rol. Es preciso recalcar que para el desarrollo de la calendarización se implementó la metodología scrum, es por ello que el diagrama Gantt está dividido en 3 secciones, correspondientes a los *sprints* semanales, esto quiere decir que para cada *Sprint* existen y se evidencian un número de tareas semanales con una mayor intensidad al principio del proyecto, esto con el fin de obtener un resultado de análisis y revisión más amplio al final del ejercicio.

*Trello* fue integrado con *Monday* para llevar el control de cada tarea tanto en el diagrama Gantt como en el tablero Kanban de Trello, donde puede ser organizado todo de una manera mucho más intuitiva y visual de no ser manipulada por el *Scrum* *master* o el *product* *owner*.

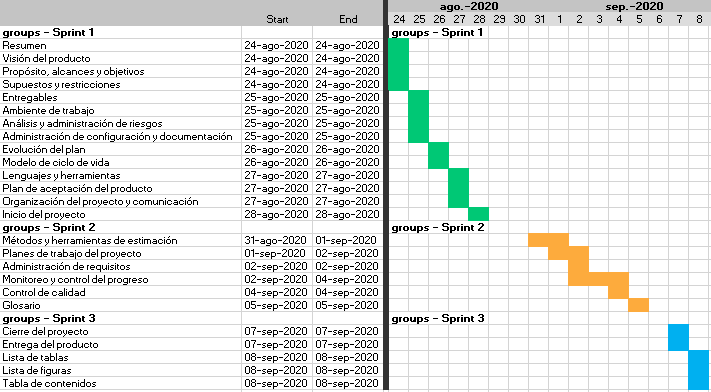




Figura 6 Diagrama de Gantt

### Presupuesto del proyecto

De acuerdo con la metodología seleccionada ([Ver sección metodología](#_Metodología_seleccionada)) se tomó como punto de análisis para el proyecto en cuanto a la estimación de costos el total de horas laboradas en cada sprint por miembro, esto tomando en cuenta la responsabilidades y cantidad de responsabilidades a desempeñar, para ello se designó un salario base tomado como consideración salarios similares investigados según roles acordes en Colombia como administrador de bases de datos, diseñador gráfico, desarrollador *back*/*front* *end*, analista de calidad entre otros.

Los salarios fueron investigados en páginas de anuncios laborales como indeed.com([www.indeed.com](http://www.indeed.com)) y elempleo ([www.elempleo.com](http://www.elempleo.com)), luego de ello al tener una variación dependiendo de factores como años de experiencia, habilidades dominadas y tiempo en el cargo se realizó un promedio por labor o cargo y se procedió a realizar los respectivos cálculos.

Para el desglose de actividades y costos por sprint, es decir por horas de trabajo semanal, se tomó como base el trabajo horario por mes establecido en Colombia, es decir 205 horas según [52], esto como resultado dio valores representativos de un costo bastante acorde al rol de cada miembro [ver anexo presupuesto](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Presupuestos.xlsx) y se identificó en qué Sprint se produjo una mayor fluctuación monetaria.

Al final se determinaron posibles imprevistos y sus costos en cuanto a la cantidad de horas invertidas para así generar un total más acorde a la realidad del proyecto puesto en marca [ver anexo presupuestos](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Presupuestos.xlsx) y con ello se determinó el costo total en cuanto a contratación y roles, ya que para esta entrega el costo de software no es contemplable al estar utilizando software gratuito o de modalidad limitada.

# Monitoreo y control del proyecto

Esta sección tiene el propósito de describir cuáles serán las herramientas y métodos que se utilizarán para monitorear el desarrollo del proyecto. Esto será de gran ayuda para que los administradores del proyecto tomen las decisiones correctas al tener en cuenta los límites de tiempo asignados a cada tarea. Adicional a esto, contiene una sección para la cual se han establecido unos pasos a seguir en caso de que se presente algún problema durante el desarrollo del proyecto.

## Administración de requisitos

El procedimiento a seguir para reportar, detectar, y/o controlar los cambios que tendrán los requisitos del producto es el siguiente. En dado caso que el *Product Owner* considere necesario realizar cambios a los requisitos del producto, este deberá informar al *Scrum Team,* esto con el propósito de que ellos estudien la forma en la que estos cambios impactaran el proyecto. Así mismo, determinaran la factibilidad de estos cambios. En caso de que dichos cambios no sean factibles, se debe especificar la razón y cuáles serían los cambios para lograr un impacto positivo con el cambio de los requisitos del producto. Una vez el *Scrum Team* haya terminado de discutir lo antes mencionado, deberán generar y enviar un reporte al *Product Owner* para que revise y decida sobre las propuestas de requisitos del producto.

A continuación se encuentra el diagrama BPMN de este proceso.

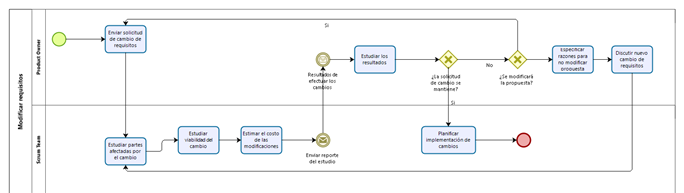


Figura 7 Diagrama BPMN Modificar Requisitos

## Monitoreo y control del progreso

Esta sección tiene el propósito de definir la rúbrica que se utilizará para medir el progreso del proyecto. Además, describe la forma en que los integrantes del grupo reportarán el trabajo realizado.

### Medidas del proyecto

Un proyecto se define como una planificación de una serie de objetivos que, al cumplirlos, se obtiene el producto final que se espera. La palabra planificación es clave pues existe un tiempo de entrega el cual se debe cumplir. Por este motivo es indispensable contar con un cronograma del proyecto el cual tendrá explícitamente las tareas a llevar a cabo con un horario estricto a cumplir.

“Un conjunto de tareas es una colección de tareas de trabajo de ingeniería del *software*, hitos, productos operativos y filtros de aseguramiento de la calidad que deben lograrse para completar un proyecto particular”[43].El proyecto BuenVecino se compone de un grupo de trabajo, fecha límite de entrega y un conjunto de tareas; esto supone que el progreso del proyecto estará basado mediante la técnica de diagrama de Gantt. Es decir, el avance estará determinado por el número de tareas realizadas en el tiempo límite establecido.

### Actividades para realizar

Se llevará control de las actividades utilizando la herramienta *Monday*. Esta página web permite crear una cuenta totalmente gratis para realizar proyectos con la condición de ser un estudiante universitario. Este plan cuenta con un límite de máximo 100 tareas para construir el diagrama de Gantt, lo cual es suficiente para el proyecto BuenVecino. Se escogió *Monday* por ser gratuita, además, posee una interfaz interactiva pero precisa, esto facilita la gestión de tareas y por ende nos permite desarrollar el proyecto a un mejor ritmo. En *Monday* estarán las fechas de inicio y cierre de cada tarea, estatus de la tarea y el responsable de cada tarea.

Con el fin de llevar un mejor control del cronograma de las actividades para realizar, se discutieron algunas recomendaciones dentro del grupo y se llegó a lo siguiente:

1. Comprobar que cada miembro del grupo cumplió con la fecha de inicio planeada para cada tarea del proyecto. Si algún miembro incurrió en alguna falta, se deberán tomar las respectivas acciones correctivas mencionadas a continuación.
2. Hacer reuniones semanales para discutir el estado del proyecto. Las reuniones siempre se anunciarán con tres días de anticipación para concordar una fecha en la que todos los miembros del grupo puedan asistir por un tiempo mínimo de sesenta minutos. [43]ingenie

### Acciones correctivas

A continuación, se mostrarán las acciones correctivas que se tomarán cuando algún miembro grupo incumpla las normas establecidas. Las sanciones impuestas funcionaran mediante un sistema de puntuación. Todos los miembros del grupo iniciarán con un puntaje de 5.0 y dependiendo de la gravedad de la falta que cometa, su puntaje se verá afectado. Si un miembro del grupo comete reiteradas faltas hasta obtener un puntaje de 3.0, este deberá tener una reunión con los demás miembros del grupo. En dado caso que su puntaje baje a 2.5, el *Scrum Master* deberá comunicarle esta situación al profesor encargado de la materia. Si el *Scrum Master* llegará a obtener un puntaje de 4.0, se llevaría a cabo una reunión con los demás miembros del grupo para discutir su continuidad como *Scrum Master*.

|  |  |  |
| --- | --- | --- |
| **Falta** | **Detalle** | **Sanción** |
| Poca comunicación | Comportamiento reiterado. El miembro del equipo no responde al llamado de los otros integrantes. | Deducción de 0.3 puntos al total del puntaje. |
| Tarea no terminada | Se le asigna una tarea a un miembro del equipo y este no termina el 100% de esta dentro del tiempo esperado. | Deducción de 0.3 puntos al total del puntaje. |
| Tarea incompleta | El miembro del equipo termina su tarea, sin embargo, esta cuenta con algunos errores. | Llamado de atención. Deducción de 0.3 puntos si el comportamiento es reiterativo. |
| Tarea no iniciada | El miembro del equipo no realiza la tarea que se le asigno. | Deducción de 1.0 puntos al total del puntaje. |

Tabla 9 Acciones correctivas

Con estas acciones correctivas se espera establecer un mecanismo eficaz para tratar cualquier tipo de inconvenientes dentro del grupo de trabajo. A continuación, se mencionará la técnica a utilizar para continuar con el desarrollo del proyecto si se llegará a presentar alguna de las faltas mencionadas anteriormente.

* ***Time-boxing:*** esta estrategia es oportuna cuando se asume que el producto final no podrá ser entregado a tiempo a nuestro cliente. Para esto se trata de dar un tiempo extra a aquellas tareas que presenten atrasos durante el desarrollo del proyecto, implementando las partes más cruciales para el desarrollo del proyecto y posponiendo las de menor importancia.[43]

## Cierre del proyecto

Para llevar a cabo el cierre del proyecto se utilizará la metodología de reporte gerencialel cual permite un mejoramiento continuo en base a las experiencias que deja la realización de este proyecto de *software*. El objetivo del cierre del proyecto es evaluar el desempeño del grupo de trabajo y los problemas que se presentaron durante el desarrollo del producto final, para de esta manera mejorar en próximos proyectos.

**Criterios de entrada:**

El desarrollo del producto final ha concluido y se han hecho las respectivas pruebas. El grupo de trabajo posee información en cuanto al desempeño durante el trabajo realizado.

**Revisión de los datos del proceso:**

* Identificar las posibles fallas durante el desarrollo del proyecto
* Comparar la planeación previa al proyecto *versus* el verdadero desempeño una vez terminado el producto
* Evaluar los datos que utilizó cada miembro durante la realización del trabajo.

Al concluir esta revisión obtendremos los problemas que surgieron a lo largo del desarrollo del proyecto, esto con el fin de prepararnos para afrontar estos problemas en un futuro.

**Revisión de los datos de calidad:**

* Evaluar la calidad del producto.
* Comparar con reportes gerenciales anteriores.
* Preparar el *Personal Investment Performance*.

En esta subsección se requiere identificar si los criterios usados para garantizar la calidad del proyecto podrían mejorar. Además, revisa si las recomendaciones propuestas en informes gerenciales pasados han sido aplicadas o si se necesitan de nuevas modificaciones para un proyecto futuro.

**Evaluación de roles:**

* Evaluar el trabajo realizado por cada miembro de acuerdo con su rol establecido.

En esta sección se evaluará el desempeño de cada miembro del grupo que tuvo dentro del proyecto. Para esto se estará revisando el trabajo que cada miembro realizó, los recursos utilizados, los problemas que aparecieron durante el desarrollo del proyecto, los aspectos que podrían mejorar a futuro y de qué manera se implementarían. La evaluación de roles está a cargo del líder del proyecto, en este caso el *Scrum Master*.

**Reporte del ciclo:**

* Informar en que partes durante el desarrollo del proyecto se presentaron o no fallas
* Resumir el desempeño de cada miembro al desempeñar su rol asignado
* Explicar cuáles fueron los resultados, el proceso y roles que se utilizaron para obtener dichos resultados.

Esta sección sirve como una conclusión al desempeño general del proyecto, la cual debe ser justificada con la información generada al desarrollar el producto final. [53]

**Sprint Review:**

* Revisar las tareas completadas y no completadas.
* Opiniones acerca del sprint recién superado.
* Realizar una mejora continua del proceso de trabajo.
* Cuatro horas como límite.

**Retrospectiva del sprint:**

El sprint número uno contó con catorce tareas en un lapso de cinco días, evidentemente esto fue un error pues no se distribuyeron bien el número de tareas a comparación de los demás sprint. Esto conllevo a que no se pudieran completar todas las tareas dentro del tiempo estipulado para el sprint 1. Estas tareas fueron terminadas antes de iniciar con el sprint pues existía un margen de tiempo que permitía finalizar dichas tareas antes de empezar a trabajar con el sprint 2. Como consecuencia, el grupo tomó la decisión de estimar mejor los tiempos y tareas de cada sprint para futuras entregas. Como ya se habían completado una buena parte de las tareas para la entrega uno, los sprint dos y tres fueron completados dentro del tiempo estipulado y esto dio tiempo para revisar y pulir las secciones más importantes del documento.

# Entrega del producto

El sitio web producto del proyecto será montado en un servidor de hosting gratuito (que por el momento no ha sido definido) con el fin de que los usuarios que deseen acceder lo hagan desde cualquier parte. Obviamente este servicio no tendrá los beneficios que un servicio de *hosting* pago o un servidor propio tendría, pero es suficiente para realizar las demostraciones de funcionamiento. Otra de las razones para hacer esto es para poder usar un nombre de dominio sin necesidad de pagar por ello, dado que este proyecto no está pensado con el fin de generar ganancias monetarias. Por otra parte, solo se garantizará el perfecto acoplamiento del sitio web con el navegador Chrome, para los otros navegadores no se garantizará su correcto funcionamiento en la parte visual (*frontend*).

No se entregarán manuales de usuario ni similares, dado que la ayuda será suministrada de forma interna dentro del sitio web. Por otra parte, el código fuente del software será publicado en la plataforma *Github*, para que pueda ser consultado de manera libre.

# Procesos de soporte

En esta sección se mostrarán los procesos de soporte del proyecto, los cuales, funcionan como permitirán la correcta ejecución de este.

## Ambiente de trabajo

Se establecieron reglas generales, que cada integrante del grupo debe cumplir, para generar un ambiente de trabajo ideal dentro de este. De tal manera, se generaron anexos ([ver anexo actas](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Actas)), en donde se documenta el progreso con respecto al proyecto, se tiene un registro de los participantes, con el objetivo de poder lograr las metas en el tiempo establecido y mejorar la productividad a lo largo del proyecto. Para esto, se acordaron sanciones por infracciones cometidas, para llevar registro de la falta de las reglas, al iniciar las reuniones se evaluará si hubo algún tipo de rompimiento a alguna de ellas y se procederá a colocar una sanción si es necesario. Las reglas establecidas se describen a continuación.

### Reglas de trabajo

1. El principal medio de comunicación será el grupo de *WhatsApp* y debido a la virtualidad se realizan reuniones vía *Microsoft* *Teams* una vez por semana para cada *Sprint* y algunas en horarios extraoficiales.
2. En las reuniones semanales estará prohibido tocar temas que no estén relacionados con el proyecto.
3. Las reuniones semanales son de carácter obligatorio, por lo que todos los integrantes del grupo deberán asistir de manera puntual a cada una de ellas.
4. Cada miembro del equipo debe ser responsable por las tareas que le fueron asignadas cada semana.
5. Cada miembro del equipo debe dirigirse a los demás con respeto.
6. Los problemas personales del equipo deben ser resueltos sin que perjudiquen el avance del proyecto.
7. La inasistencia a una reunión debe ser justificada por medio de una excusa válida.
8. La inasistencia injustificada a una reunión se considerará como una posible causal de expulsión del grupo.
9. Las citaciones se realizarán en el formato *IEEE*, es importante que todos los miembros del equipo usen este formato al realizar sus citas.

### Excusas válidas

Ante una ausencia a una reunión, el integrante deberá presentar una excusa válida, para el equipo de *ECO* dichas excusas serán las siguientes:

1. Emergencia médica.
2. Cita médica programada.
3. Razones personales como una tragedia familiar.
4. Enfermedad que no le permita la asistencia y normal participación dentro de la reunión.

### Sanciones y acciones correctivas

Si uno de los miembros de *ECO* incurre en una falta a alguna de las reglas se procederá a una reunión del equipo y determinar la sanción correspondiente, éstas pueden ir desde llamados de atención en frente de todo el grupo hasta la expulsión del integrante del equipo de trabajo, depende de la concurrencia a estas faltas.

## Análisis y administración de riesgos

Se realizó un análisis de riesgos con el fin de identificar aquellos que de alguna forma amenazan al proyecto o su correcto desarrollo dentro del contexto planteado por el grupo de trabajo.

### Proceso de identificación de riesgos

Para identificar cada uno de los riesgos encontrados en el proyecto se siguió un proceso sistemático que permitió identificar, clasificar, encontrar causas, analizar posibles consecuencias y validar opciones que redujeran los impactos negativos de los riesgos sobre el proyecto. Los resultados de estos análisis fueron plasmados en el documento de Matriz de Riesgos ([ver anexo tabla matriz de riesgos](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Matriz%20de%20riesgos.xlsx)) donde se explica cada uno de los resultados de este proceso. Este proceso fue realizado en el inicio del proyecto, sin embargo, será un proceso continuo que se seguirá ejecutando largo del desarrollo general del proyecto, con el fin de encontrar nuevos riesgos que puedan aparecer en las siguientes etapas o que hayan sido pasados por alto en esta primera etapa. Este proceso es descrito de forma más precisa en el diagrama BPMN de proceso de identificación de riesgos.

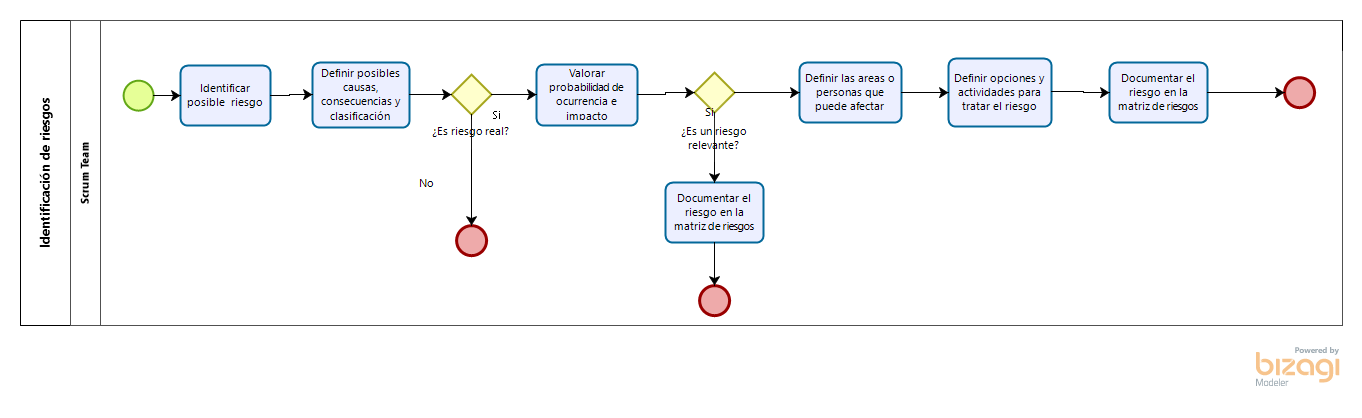


Figura 8 Diagrama BPMN Identificación de Riesgos

### Clasificación de riesgos

Con el fin de tener mayor control y conocimiento sobre los riesgos se realizó una clasificación acorde a su origen de la siguiente forma:

* Interno: si el riesgo es propio del equipo de trabajo
* Externo: si el riesgo proviene de un actor que no pertenece al equipo de trabajo
* Tecnológico: si el riesgo es propio de la tecnología usada

Adicionalmente se identificaron las causas que hacían que el riesgo apareciera y las consecuencias que este podría traer sobre el desarrollo del proyecto. Por último, se definió una escala de valores con el fin de cuantificar los riesgos. Esta escala es aplicada a dos variables que poseen valores numéricos y van desde 1 hasta 5, y son interpretados de la siguiente forma:

* La probabilidad de que el riesgo ocurra, siendo 1 muy poco probable y 5 altamente probable
* El impacto que puede tener en caso de llegar a suceder, siendo 1 un impacto muy pequeño (pero lo suficiente para ser tenido en cuenta) y 5 un impacto que con gran probabilidad impedirá la realización del proyecto (aunque no lo convierte en imposible de realizar)

El valor del riesgo se definió entonces como , siendo así el valor mínimo 1 donde el riesgo está presente pero no representa mayor peligro, y 25 el valor máximo, donde el riesgo representa un peligro bastante grande para el proyecto.

### Tratamiento del riesgo

Para cada riesgo identificado se definieron estrategias para tratarlo para evitar que el riesgo se dé o en su defecto hacer que su impacto negativo sobre el proyecto sea lo más pequeño posible, junto con los responsables de hacerlo y los recursos involucrados. Las formas de tratar un riesgo fueron definidas de la siguiente forma:

* Evitar: cuando la estrategia se enfoca en impedir que el riesgo se materialice haciendo el mayor esfuerzo posible. Estas estrategias serán ejecutadas de forma continua durante la realización del proyecto ya que el riesgo permanece durante todo el desarrollo del proyecto.
* Mitigar: cuando la estrategia se enfoca en solucionar el problema en caso de que el riesgo se materialice, dado que evitar que el riesgo suceda es muy complicado (esto no significa que no se vayan a dedicar esfuerzos a evitar que suceda). Estas estrategias por su parte serán ejecutadas cuando el riesgo se dé (en caso de que se llagase a dar), ya que es muy difícil predecir cuándo sucederá esto.

Dichas estrategias y sus responsables de ejecutarlas son descritas para cada riesgo identificado el documento de Matriz de Riesgos ([ver anexo matriz de riesgos](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Matriz%20de%20riesgos.xlsx))

### Riesgos identificados

Fueron identificados un total de diez riesgos relevantes, aunque también se sabe de la presencia de otros posibles riesgos que, debido a su baja probabilidad, poca relevancia o nula capacidad de control sobre ellos, no fueron tenidos en cuenta para el análisis. Entre los diez riesgos considerados como relevantes destacan los asociados al posible fracaso en el aprendizaje y uso de las tecnologías escogidas por parte del equipo de trabajo, dado que ningún integrante ha trabajado previamente con dichas tecnologías, razón por la cual el equipo se enfrenta a distintos retos al tiempo, como el aprendizaje de la tecnología y desarrollo del *software*, además de tener tiempo limitado para lograrlo. Aunque este riesgo es el más relevante, se ha decido enfocar los mayores esfuerzos (como individuos y como grupo) con el fin de superarlo.

Estos y otros riesgos son descritos con mayor profundidad teniendo en cuenta los criterios anteriormente descritos en la matriz de riesgos ([ver anexo matriz de riesgos](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Matriz%20de%20riesgos.xlsx))

## Administración de configuración y documentación

|  |  |  |
| --- | --- | --- |
| **Ítem de configuración** | **Descripción** | **Momento de creación y modificación** |
| WBS  (Work Breakdown Structure) | Es una estructura utilizada para descomponer analíticamente un proyecto en partes elementares.  El objetivo es organizar el trabajo en elementos fáciles de manejar y volver menos complicada la comprensión del proyecto.[54] | Creado en la primera entrega del proyecto, puede ser modificado únicamente por todo el grupo en las reuniones semanales. |
| SPMP (Software Project Management Plan) | Es un documento bien organizado que contiene la planificación del proyecto en detalle.[55] | Puede ser modificado si el Scrum decide hacer cambios en la planificación o si es necesario realizar correcciones solicitadas por el profesor del curso. |
| Diagrama de Gantt | Es una herramienta útil para planificar proyectos. Al proporcionar una vista general de las tareas programadas, todas las partes implicadas sabrán qué tareas tienen que completarse y en qué fecha.[56] | Creado desde la primera entrega. Puede ser modificado por el encargado del monitoreo en este proyecto cuando se realizan las reuniones semanales. |
| Códigos fuente | Archivos en los cuales se escribe el código del proyecto | Creado luego de haberse realizado la primera entrega. Todos los integrantes pueden modificarlo en cualquier momento. |
| Tabla de riesgos | Tabla con la identificación, con la probabilidad de ocurrencia e impacto. | Creado desde la primera entrega |

Tabla 10 Ítems de configuración

El proceso de control de cambios a los ítems de configuración incluye dos tipos:

* + - 1. Relacionado a las funcionalidades: son aquellos que se relacionan con los cambios de requisitos. Esto es, que afecta a los artefactos de diseño como a los códigos fuente del producto.
      2. Relacionado a la documentación: son aquellos en comunicación al ciclo de vida y que pueden ser modificados, por ejemplo, se encuentra la tabla de riesgos o el diagrama de Gantt.

Esto comienza cuando el *Scrum* *Team* ha identificado un cambio, y debe clasificarse en uno de los dos tipos anteriormente mencionados. Si el cambio pertenece a funcionalidad se debe realizar el procedimiento de *Modificar requisitos* especificado en la sección 9.1 de este documento. Finalmente, el paso a seguir es redactar un breve resumen del cambio del requisito en el acta de la reunión.

Si el cambio pertenece a documentación, este se añade a la tabla de cambios y se registra en el acta de la reunión.

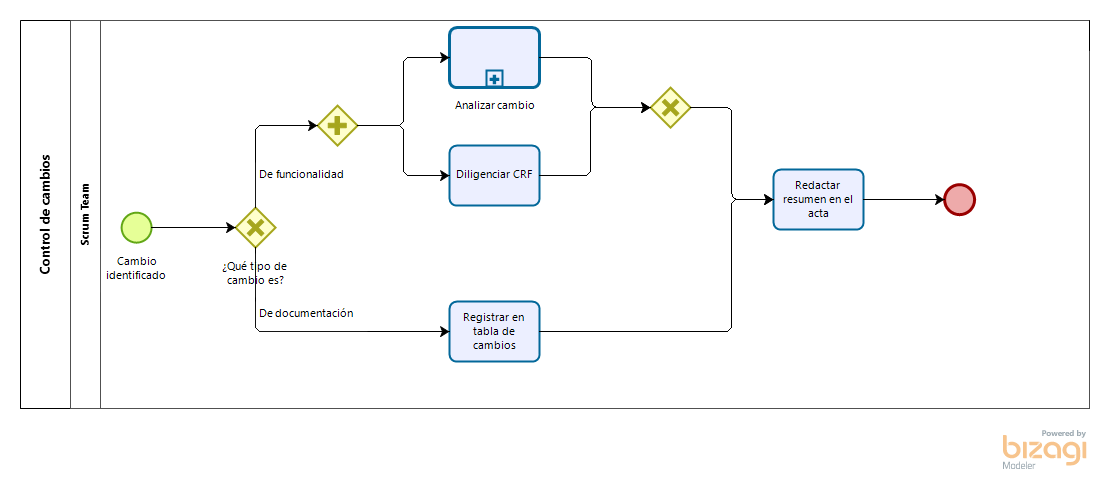


Figura 9 Diagrama BPMN control de cambios

Por último, este proceso se lleva a cabo si se realiza el cambio o no. Si el cambio es de funcionalidad esta pregunta se hace en el proceso de analizar cambio, por otro lado, si es de documentación, el ítem de configuración será modificado y se debe hacer registro en el acta, ya dependerá del grupo de trabajo si los cambios se mantienen o no.

## Control de calidad

### Medidas de Calidad:

Debido a se encuentran múltiples características que se deben cumplir para tener un proyecto de alta calidad, tanto en el ámbito de la documentación como en el ámbito de desarrollo de *software*, se ha decido realizar una estrategia que permita la flexibilidad al momento de determinar los factores de la calidad para una determinada tarea. Ya que se está utilizando una metodología ágil como Scrum que permite las revisiones constantes con un efecto menor en la productividad del proyecto.

Para la medida de calidad para el *software* se tomará en respecto a los factores de calidad del *ISO 25010*, que son: adecuación funcional, eficiencia de desempeño, compatibilidad, usabilidad, fiabilidad, seguridad, mantenibilidad y portabilidad[57]. En la parte de la documentación, hacemos referencia a las características a cumplir presentadas por el profesor del curso de Ingeniería de *software*.

En otras palabras, la calidad de las tareas está reflejada respecto al número de características que el grupo identifica para cada una de estas, con las características que fueron implementadas correctamente.

### Proceso para control de calidad de los documentos

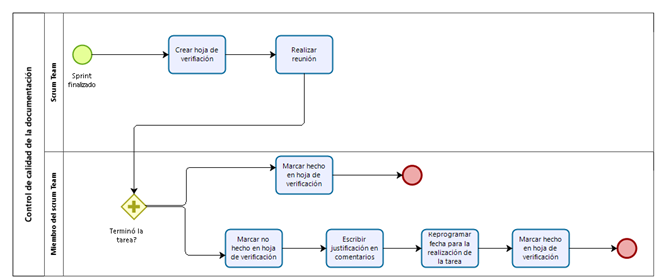


Figura 10 Diagrama BPMN control de calidad de la documentación

El proceso comienza en las reuniones del *Scrum Team* al momento de definir las tareas de cada Sprint, el grupo de trabajo comenzará a desarrollar un plan de gestión de calidad, donde se darán las características que debe tener cada una de las secciones de la documentación al momento de realizar cada Sprint. Dado esto se generará una hoja de verificación con las características dadas, que posteriormente se diligenciará.

Cuando finaliza el sprint en la reunión semanal que hace el *Scrum Team* se diligenciará la hoja de verificación. Si las tareas no fueron completadas, se debe documentar la razón con el fin de que esta tarea se ponga en el siguiente *Sprint* y que el problema sea resuelto.

### Proceso para control de calidad de software:

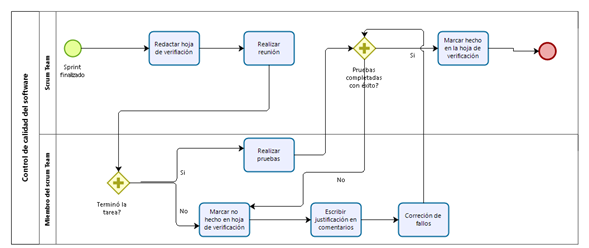


Figura 11 Diagrama BPMN control de calidad del Software

Cuando se ha terminado de redactar el *Sprint* *Backlog* hecho por el *Scrum Team*, el grupo debe redactar una hoja de verificación con la definición de las tareas con sus respectivas características para sean completadas.

Al realizar la reunión semanal y terminar el *Sprint*, cada uno de los miembros del grupo debe diligenciar la hoja de verificación. Ya si un miembro no ha terminado su tarea asignada, en las conclusiones debe explicar el por qué, para que la tarea sea resuelta en el próximo *Sprint*

Si la tarea asignada fue completa y se demuestra que a esta se le han realizado pruebas de su funcionamiento, el *codeline* se añade al *baseline*.

# Anexos

1. Actas (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Actas))
   1. Acta 1 (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Actas/Acta1.docx))
   2. Acta 2 (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Actas/Acta2.docx))
   3. Acta 3 (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Actas/Acta3.docx))
   4. Acta 4 (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Actas/Acta4.docx))
2. Miro (Acceso en: [Enlace](https://miro.com/app/board/o9J_kncNyAM=/))
3. Test Hermann (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Test%20Herrmann))
4. Presupuesto (Acceso en: [Enlace](https://livejaverianaedu-my.sharepoint.com/:x:/g/personal/j_afanador_javeriana_edu_co/EdA3lufOtGdFinls_7qsApEBYJBixP_h-VRL4uKD0SAxAg?e=Z7XaCa))
5. Riesgos (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Matriz%20de%20riesgos.xlsx))
6. Hojas de verificación (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Hojas%20de%20Verificación))
7. Hoja de verificación 1 (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Hojas%20de%20Verificación/Hoja%20de%20verificación%201.docx))
8. Hoja de verificación 1 (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Hojas%20de%20Verificación/Hoja%20de%20verificación%202.docx))
9. Hoja de verificación 1 (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Hojas%20de%20Verificación/Hoja%20de%20verificación%203.docx))
10. Reporte gerencial (Acceso en: [Enlace](https://livejaverianaedu.sharepoint.com/sites/Ingesoft878/Shared%20Documents/General/Anexos/Reporte%20Gerencial.docx))

# Referencias

[1] “Definición de arrendatario - Qué es, Significado y Concepto.” https://definicion.de/arrendatario/ (accessed Sep. 09, 2020).

[2] “Definición de Arrendamiento » Concepto en Definición ABC.” https://www.definicionabc.com/derecho/arrendamiento.php (accessed Sep. 09, 2020).

[3] “Geolocalización: virtudes y riesgos | Oficina de Seguridad del Internauta.” https://www.osi.es/es/actualidad/blog/2016/09/20/geolocalizacion-virtudes-y-riesgos (accessed Sep. 09, 2020).

[4] “Significado de Stakeholder (Qué es, Concepto y Definición) - Significados.” https://www.significados.com/stakeholder/ (accessed Sep. 09, 2020).

[5] “¿Qué es un framework y para qué se utiliza? | Orix Systems.” https://www.orix.es/que-es-un-framework-y-para-que-se-utiliza (accessed Sep. 09, 2020).

[6] “Qué es un Sprint de Scrum | OpenWebinars.” https://openwebinars.net/blog/que-es-un-sprint-scrum/ (accessed Sep. 09, 2020).

[7] “¿Qué Es Un Hosting? Hosting Web Explicado Para Principiantes.” https://www.hostinger.co/tutoriales/que-es-un-hosting?\_\_cf\_chl\_jschl\_tk\_\_=b7520162c2a6ac4c2cd7f2026b0eeca1dad8305d-1599690731-0-AWYi2fewdGQh1HZ6UmjlLk3jXklqBQexuVth8qh4iBM4Pho6g-7V3HUGE\_4TLF-o8rFdBWG7BAxobgGpA-Whnlx5D27C5LGR0FUnGJYdTJF-1Fj7JIO33heGJL0Js1MY6D3k\_uLi9TP2xpBHSIhZYtf5dzN0q6CFai4325jVzZIVUgSoo4z\_kO9iMEObmQFU-5OARBGkcVv4nR2hd8PZQFba0VNXnMnHV1CUi8aI2qR1UaW6XPhMpciasCx8KZuok1X5l87kFC\_2D7LO\_f7TzVct2x5s54Tl0\_A7SnRE4S8sIiqbvLL4ooG1wovIj8k7OA (accessed Sep. 09, 2020).

[8] “Biblioteca (informática) - Wikipedia, la enciclopedia libre.” https://es.wikipedia.org/w/index.php?title=Biblioteca\_(informática)&oldid=126104731 (accessed Sep. 09, 2020).

[9] L. G. Definitiva, “La Guía de Scrum TM.” Accessed: Aug. 24, 2020. [Online]. Available: http://creativecommons.org/licenses/by-sa/4.0/legalcodeandalsodescribedinsummaryformathttp://creativecommons.org/licenses/by-sa/4.0/.Byutilizing.

[10] “What is Extreme Programming (XP)? | Agile Alliance.” https://www.agilealliance.org/glossary/xp/#q=~(infinite~false~filters~(postType~(~’post~’aa\_book~’aa\_event\_session~’aa\_experience\_report~’aa\_glossary~’aa\_research\_paper~’aa\_video)~tags~(~’xp))~searchTerm~’~sort~false~sortDirection~’asc~page~1) (accessed Aug. 24, 2020).

[11] “Kanban: Control de materiales y producción - Ingenieria Industrial Online.” https://www.ingenieriaindustrialonline.com/lean-manufacturing/kanban-control-de-materiales-y-produccion/ (accessed Aug. 24, 2020).

[12] “Metodología Kanban: revoluciona tu manera de trabajar más ágil.” https://blog.trello.com/es/metodologia-kanban (accessed Aug. 24, 2020).

[13] “Metodología Kanban | Kanban Tool.” https://kanbantool.com/es/metodologia-kanban (accessed Aug. 24, 2020).

[14] “Documentation for Visual Studio Code.” https://code.visualstudio.com/docs (accessed Sep. 09, 2020).

[15] “Best of Atom: Features, Plugins, Acting Like Sublime Text ― Scotch.io.” https://scotch.io/bar-talk/best-of-atom-features-plugins-acting-like-sublime-text (accessed Sep. 09, 2020).

[16] “HTML: Lenguaje de etiquetas de hipertexto | MDN.” https://developer.mozilla.org/es/docs/Web/HTML (accessed Sep. 09, 2020).

[17] “Bootstrap 4 Get Started.” https://www.w3schools.com/bootstrap4/bootstrap\_get\_started.asp (accessed Sep. 09, 2020).

[18] “The most advanced responsive front-end framework in the world. | Foundation.” https://get.foundation/ (accessed Sep. 09, 2020).

[19] “facebook/react: A declarative, efficient, and flexible JavaScript library for building user interfaces.” https://github.com/facebook/react (accessed Sep. 09, 2020).

[20] “Introduction — Vue.js.” https://vuejs.org/v2/guide/ (accessed Sep. 09, 2020).

[21] “Angular - Introduction to Angular concepts.” https://angular.io/guide/architecture (accessed Sep. 09, 2020).

[22] “All you need to know about Firebase.” https://www.the-reference.com/en/blog/pieter-de-busschere/firebase (accessed Sep. 09, 2020).

[23] “Bases de datos de Azure: servicios en la nube totalmente administrados | Microsoft Azure.” https://azure.microsoft.com/es-es/product-categories/databases/ (accessed Sep. 09, 2020).

[24] “What is Git? - Azure DevOps | Microsoft Docs.” https://docs.microsoft.com/en-us/azure/devops/learn/git/what-is-git (accessed Sep. 09, 2020).

[25] “What is SVN? | Subversion Overview | Perforce.” https://www.perforce.com/blog/vcs/what-svn (accessed Sep. 09, 2020).

[26] “What is GitHub.” https://www.w3schools.com/whatis/whatis\_github.asp (accessed Sep. 09, 2020).

[27] “What is GitLab? | GitLab.” https://about.gitlab.com/what-is-gitlab/ (accessed Sep. 09, 2020).

[28] “What is StarUML - StarUML Documentation.” https://documentation.help/StarUML/what\_is\_staruml.htm (accessed Sep. 09, 2020).

[29] “Visual Paradigm Frequently Asked Questions.” https://www.visual-paradigm.com/support/faq.jsp (accessed Sep. 09, 2020).

[30] “Bizagi Oficial Plataforma de negocios digitales y BPMS.” https://www.bizagi.com/es/plataforma/modeler (accessed Sep. 09, 2020).

[31] “What Is G Suite? How To Use Google Suite Products | Bluehost Support.” https://www.bluehost.com/help/article/gsuite (accessed Sep. 09, 2020).

[32] “What is Microsoft Office 365? Webopedia Definition.” https://www.webopedia.com/TERM/O/office\_365.html (accessed Sep. 09, 2020).

[33] “¿Qué es Trello? - Ayuda de Trello.” https://help.trello.com/article/708-what-is-trello (accessed Sep. 09, 2020).

[34] “What is monday.com? – Support.” https://support.monday.com/hc/en-us/articles/115005310945-What-is-monday-com- (accessed Sep. 09, 2020).

[35] I. Sommerville, *GlobAl EdiTioN Software Engineering TENTH EdiTioN*. 2016.

[36] R. S. Pressman, *Ingenieria del Software. Un Enfoque Practico*. 2010.

[37] “Applied Scrum for Agile Project Management | edX.” https://www.edx.org/course/usmx-applied-scrum-for-project-management-2?utm\_medium=affiliate\_partner&utm\_source=coursearena-coursearena (accessed Sep. 09, 2020).

[38] “Scrum Foundations Professional Certificate (SFPC) - (Spanish) | CertiProf.” https://certiprof.com/pages/scrum-foundations-professional-certificate-sfpc-spanish (accessed Sep. 09, 2020).

[39] “React JS ⭐Desde Cero! Hooks, Redux, Context, Firebase y más! | Udemy.” https://www.udemy.com/course/curso-react-js/ (accessed Sep. 09, 2020).

[40] “What is Microsoft Word?” https://www.computerhope.com/jargon/m/microsoft-word.htm (accessed Sep. 09, 2020).

[41] J. Gelfand, “Research Guides: Mendeley Research Manager: Why Mendeley?,” Accessed: Sep. 09, 2020. [Online]. Available: https://guides.lib.uci.edu/Mendeley/why\_mendeley.

[42] “What is Planning Poker®? – Planning Poker®.” https://www.planningpoker.com/answer/what-is-planning-poker/ (accessed Sep. 09, 2020).

[43] “Inicio - Canva.” https://www.canva.com/ (accessed Sep. 09, 2020).

[44] “11vo Coloquio WBS Erick Oliveros.” https://pdfslide.net/business/11vo-coloquio-wbs-erick-oliveros.html (accessed Sep. 09, 2020).

[45] “¿Una empresa puede extender la jornada laboral a más de 8 horas? - Servicios - Justicia - ELTIEMPO.COM.” https://www.eltiempo.com/justicia/servicios/una-empresa-puede-extender-la-jornada-laboral-a-mas-de-8-horas-425372 (accessed Sep. 09, 2020).

[46] W. S. Humphrey, “The Team Software Process SM (TSP SM ),” 2000.

[47] “WBS (Work Breakdown Structure), qué es y cómo se usa - BibLus.” http://biblus.accasoftware.com/es/wbs-workbreakdownstructure-que-es-y-como-se-usa/ (accessed Sep. 09, 2020).

[48] “Software Project Management Plan - SPMP Document.” https://tutorialsinhand.com/tutorials/software-engineering-tutorial/software-project-management/spmp-document.aspx (accessed Sep. 09, 2020).

[49] “¿Qué es y para qué sirve un diagrama de Gantt?” https://blog.teamleader.es/diagrama-de-gantt (accessed Sep. 09, 2020).

[50] “ISO 25010.” https://iso25000.com/index.php/normas-iso-25000/iso-25010 (accessed Sep. 09, 2020).