
**Integrantes:**

Barajas Diego
Cruz Brandonn
González Juan Sebastián
Miranda Nicolás
Orozco Juan David

*Healthy Routine*

Pontificia Universidad Javeriana
Ingeniería de Sistemas
Ingeniería de Software


**Docente**: Anabel Montero

Versión 1.0

# 1 Historial de cambios

| Fecha | Versión | Detalles | Encargado |
|---|---|---|---|
| 09/02/2019 | 1.0 | Creación de documento y asignación los roles. | Todos los miembros. |
| 08/02/2019 | 1.1 | Elaboración diagrama de Gantt. | Nicolás Miranda |
| 08/02/2019 | 1.2 | Administración de requisitos | Brandonn Cruz |
| 09/02/2019 | 1.3 | Elaboración del resumen y la introducción del documento. | Juan Sebastián González |
| 11/02/2019 | 1.4 | Desarrollo de la visón, alcance, objetivos y propósito, supuestos y restricciones. | Juan Sebastián González |
| 11/02/2019 | 1.5 | Inicio del contexto del proyecto, selección modelo ciclo de vida y herramientas/lenguajes.<br>Entrega del producto a futuro | Juan David Orozco<br>Diego Barajas |
| 11/02/2019 | 1.6 | Entrega del producto a futuro. | Diego Barajas. |
| 13/02/2019 | 2.0 | Compatibilidad, monitoreo y control del proyecto, cierre del proyecto. | Brandonn Cruz<br>Diego Barajas. |
| 14/02/2019 | 2.1 | Plan de aceptación del producto y administración del proyecto. | Juan Orozco |
| 15/02/2019 | 2.2 | Inicio del proyecto, planes de trabajo del proyecto y herramientas de estimación. | Nicolás Miranda. |
| 19/02/2019 | 2.3 | Revisión general del documento. Correcciones ortográficas y de redacción. Creación del glosario. | Todos los miembros. |
| 20/02/2019 | 2.3 | Análisis de riesgos (Plan de contingencia), ambiente de trabajo, evidencia gestión del proyecto. | Juan González, Brandonn Cruz y Diego Barajas. |

| | | | |
|---|---|---|---|
| 22/02/2019 | 3.0 | Administración de configuración, control de calidad, reporte gerencial. | Brandonn Cruz |
| 25/02/2019 | 3.1 | Revisión y corrección de errores en el documento | Todos los miembros |
| 20/02/2019 | 3.2 | Corrección de errores en el documento, mejoras en calidad y cohesión de las secciones 7 y 8. | Todos los miembros |

*Tabla 1 Historial de cambios*

## 2 Resumen

En el documento se presentará el proyecto *Healthy Routine*, desarrollado en la asignatura Ingeniería de Software. El aplicativo está dirigido al segmento de la población que desee llevar una rutina más saludable, enfocándose en el control de los hábitos personales por medio de rutinas físicas (*Stretching*, *Crossfit*) y dietas nutricionales acordes a las pretensiones del usuario.

Es necesario tener presente desde un principio los objetivos, el alcance y las limitaciones que tendrá el proyecto. A lo largo del documento se definirán las variables de entorno consideradas relevantes para poder llevar a cabo el proyecto satisfactoriamente; basándose principalmente en las normativas planteadas por la docente de la asignatura.

En primera instancia es importante conocer el contexto en el que se desarrollará el proyecto, elaborando así el modelo de ciclo de vida e identificando las herramientas tecnológicas principales que permitan la elaboración del trabajo entre los miembros del equipo. Por otra parte, la administración, el control y el monitoreo del plan de trabajo serán trascendentales para proveer un trabajo adecuado cumpliendo los propósitos trazados inicialmente.

El documento está dirigido a todo aquel que desee estar informado en cuánto al progreso del proyecto de Ingeniería de software haciendo énfasis en la planeación, gestión, monitoreo del trabajo desarrollado por el equipo *IngeSap.*

# 3 Tabla de contenidos

# 4 Lista de figuras



# 5 Lista de tablas

# 6 Introducción

El principal objetivo del documento es presentar y describir los procesos de negocio, el desarrollo, la organización y el funcionamiento que el equipo de trabajo alcanzará con la implementación de *Healthy Routine.*

Es relevante dar a conocer al segmento de la población interesada en el aplicativo cómo a través de la Ingeniería de Software es posible planear, desarrollar, monitorear y controlar un proyecto de ingeniería. Gracias a la documentación elaborada es posible comprender de qué manera se ha trabajado para mostrar un avance significativo en un primer escrito; permitiendo al lector profundizar en la funcionalidad que tendrá el aplicativo.

Dando un primer vistazo a la operatividad del sistema planeado, es preciso afirmar que *Healthy Routine* busca ser una aplicación móvil de interés para aquellos interesados en tener una rutina más saludable por medio de ejercicios físicos (Ejercicios aeróbicos y de flexibilidad), la regulación de la cantidad y tipo de alimentos que toma el usuario (enfoques de la nutrición física recomendados para que el usuario pueda elegir el mejor plan de dieta que se ajuste a sus objetivos).

Estas funcionalidades presentarán beneficios y ventajas para aquellos que usen la app debido a que encontrarán un sistema muy completo donde se presentará información relevante que se ajusta a sus necesidades especializándose en dos áreas principales como lo son la nutrición y el deporte. La experiencia que tendrá el usuario será el principal diferenciador del proyecto, incluyendo variedad de ejercicios para la musculación y animaciones para ver cómo practicarlos.

# 7 Vista general del proyecto

## 7.1 Visión del producto

La visión a futuro del proyecto emprendido por el equipo de trabajo es ambiciosa, se destaca la posibilidad de que el producto sea puesto en marcha como una herramienta líder en el ámbito de la salud y el deporte. Aquellos que se sientan identificados con los propósitos trazados por el aplicativo (especialmente llevar una rutina más saludable) podrán gozar de las ventajas disponibles. El proceso de desarrollo tiene una planeación estructural y organizacional que se trabajará progresivamente; definiendo de este modo 3 entregas de documentación que serán consideradas como hitos del proyecto para demostrar el avance del proyecto de Software.

## 7.2 Propósito, alcance y objetivos

**Propósito:** La intención del aplicativo móvil está centrada en la proporción de información nutricional (dietas y planes de alimentación) y física (Push up y Stretching) a los usuarios que hagan uso del sistema, dependiendo de sus hábitos y de sus intereses.
En el primer punto recomendar dietas, debido a que es importante para la salud mantener un buen control de los alimentos que se consumen, además desarrollar hábitos alimenticios saludables no es tan confuso o restrictivo como muchas personas imaginan. En segunda instancia, los planes de ejercicios físicos traen beneficios a la salud, trabajando principalmente en el humor, concentración, estrés, hábitos y demás [1].
A través de *Healthy Routine* se proporciona un sitio informativo virtual, con un diseño práctico, al cual se puede acceder a través de una amplia gama de dispositivos móviles.

**Alcance:** Diseñar un sistema que contribuya con la gestión de recomendaciones de actividad física y nutrición mediante el análisis y modelado de los procesos de negocio con un plazo máximo de 4 meses (Semestre académico). Idealmente implementando funcionalidades como:

- Las rutinas informativas que estarán enfocadas sólo en dos aspectos: enfoque nutricional y enfoque físico.
- El aplicativo permite a los administradores previamente identificados almacenar, modificar y eliminar la información correspondiente con las rutinas físicas y las dietas recomendadas.
- El usuario podrá registrarse en la aplicación y acceder a los servicios de información que ésta ofrece.
- Los administradores del sistema estarán en la capacidad de generar un reporte basado en la conectividad de los usuarios y de sus preferencias.
- El sistema ofrece información detallada de los alimentos recomendados en las dietas nutricionales.

- La plataforma no cuenta con sistema de transacciones monetarias (Pagos vía internet).
- La plataforma no cuenta con servicio de llamadas.

**Objetivos:**

**General:** Implementar un sistema práctico dirigido a personas con el interés particular de mejorar su vida saludable, haciendo énfasis en la nutrición y ejercicios físicos. De esta manera podrá elegir el mejor plan de rendimiento que se adapte a sus objetivos.

**Específicos:**

- Investigar sobre planes de nutrición y rutinas físicas.
- Procesar la información proporcionada por los usuarios del sistema.
- Proveer al usuario servicios de información de nutrición y ejercicio físico.
- Aplicar los métodos de desarrollo de proyectos vistos en el aula de clase.

## 7.3 Supuestos y restricciones

**Supuestos:**

- El grupo de trabajo estará formalmente definido por roles en áreas específicas. Los integrantes del grupo cumplen con cada uno de sus roles y tiempos de entrega.
- El trabajo desempañado es de calidad acorde a las temáticas desarrolladas en el aula de clase.
- El usuario debe tener un dispositivo móvil para poder acceder a los servicios que ofrece el aplicativo.
- La base de datos que se utilizará correrá por cuenta de Firebase, toda la información y los modelos de datos serán desarrollados en dicha plataforma.
- Es necesario que el usuario ingrese información básica de contacto e información relacionada con temas nutricionales y físicos.

**Restricciones:**

- El proyecto estará desarrollado en el primer semestre de 2019. De la misma manera el tiempo definido para cada una de las entregas del proyecto. Los límites temporales deben ser respetados tanto para la entrega de los documentos como para las presentaciones orales.

- El aplicativo estará disponible para dispositivos móviles con sistema operativo Android y iOS.
- El archivo de la aplicación será un apk.
- La aplicación no estará en la App store ni en la Play store.

## 7.4 Evolución del plan

El plan de trabajo diseñado para el desarrollo del proyecto no está exento de cambios, es evidente que a lo largo de las reuniones planeadas por el equipo de trabajo se irán contemplando las posibilidades de generar cambios en el documento. Es necesario seguir una metodología para asegurar que las variaciones se implementen correctamente sin que afecte de manera perjudicial a las demás secciones. El proceso por seguir en caso de evidenciar la necesidad de un cambio es el siguiente:

1. Identificar el evento que necesita cambios: Se establece que alguno de los integrantes del equipo de trabajo debe hacer evidente la necesidad que necesita el proyecto para cambiar alguna situación en el documento. Los demás miembros del equipo analizarán la propuesta y definirán si es necesaria una reunión grupal para acordar los cambios a realizar.

2. Reunión grupal: Si el grupo de trabajo asegura que es necesaria una reunión para aclarar puntualmente los motivos del cambio a generar y sus posibles efectos en el resto del proyecto se proseguirá a la reunión grupal. Esta reunión puede ser dentro de los encuentros que se realizan semanalmente o en su defecto en algún horario pertinente para hacer saber los detalles a todo el equipo.

3. Calificación de efectos secundarios: Se evalúan las posibilidades de cambios en cuanto a tiempo, toma de recursos y dificultan en implementar los cambios.

4. Implementación de los cambios: Posteriormente se continúa con el plan elaborado para el establecimiento de las variaciones en todas las áreas del proyecto necesarias.

5. Pos-Evaluación: Se evalúa qué tantos efectos generaron el cambio dentro del proyecto respecto a los que se plantearon en el paso 1.

Por otro lado, se considera como prioridad llevar un control de versiones del documento y del código de software que se desarrollará posteriormente. En el primer caso, las versiones se alojarán en una carpeta *Drive* compartida con el objetivo de que todos los miembros del equipo tengan acceso a los documentos pasados en caso de que sea necesario.

## 7.5 Glosario

| TÉRMINO | DEFINICIÓN |
|---|---|
| Alimentación | Es un proceso mediante al cual los seres vivos consumen diferentes tipos de alimentos para obtener de estos los nutrientes necesarios para sobrevivir y realizar todas las actividades necesarias del día a día [2]. |
| Caloría | Son producto de todos los grandes nutrientes que toda persona requiere para obtener energía; una vez que la caloría se separa de los nutrientes, estas se convierten en Kilocalorías [3]. |
| Crossfit | Programa de entrenamiento de alta intensidad y corta duración que se basa en el incremento de todas capacidades físicas básicas: Resistencia, fuerza, flexibilidad, equilibrio y agilidad [4]. |
| Dieta | Conjunto de las sustancias alimenticias que componen el comportamiento nutricional de los seres vivos [5]. |
| Fitness | Estado de salud física y bienestar que se consigue al llevar una vida sana apoyada en el ejercicio continuado en el tiempo y en una dieta saludable [6]. |
| Nutrición | La ingesta de alimentos en relación con las necesidades dietéticas del organismo [7]. |
| Proteína | Son moléculas formadas por aminoácidos que están unidos por un tipo de enlaces conocidos como enlaces peptídicos [8]. |
| Push up | Flexiones de brazo: Acción de doblar las extremidades superiores mediante la acción de los músculos [9]. |
| Resistencia | Capacidad que tiene el organismo para retrasar la fatiga [4]. |

| | |
|---|---|
| Rutina | Costumbre inveterada, hábito adquirido de hacer las cosas por mera práctica y sin razonarlas.<br>Habilidad que es únicamente producto de la costumbre [10]. |
| Stretching | Comúnmente se conoce como ejercicios de estiramiento. Busca la relajación y el fortalecimiento de los músculos [11]. |

*Tabla 2 Glosario*

# 8 Contexto del proyecto

En esta sección se encuentran los criterios para la selección de las técnicas a usar como guía a lo largo del proyecto. Los modelos de ciclo de vida considerados por el grupo de trabajo, en su totalidad fueron de metodologías ágiles porque las metodologías tradicionales carecen de características fundamentales como la versatilidad en las entregas de un software profesional, flexibilidad, adaptabilidad entre otras.

## 8.1 Modelo de ciclo de vida

### 8.1.1 Metodologías ágiles: Scrum

*Scrum* se caracteriza por ser un marco para gestión de proyectos. Los tres pilares principales del Scrum son: transparencia, inspección y adaptación [12].

- La transparencia busca mantener aspectos del proceso visible para todos los que comparten la responsabilidad de los resultados.

- La inspección se debe hacer con la frecuencia suficiente para detectar variaciones y no interrumpir la ejecución del trabajo que está siendo desarrollado, inspeccionando los prototipos y el progreso del trabajo hacia la meta del sprint.

- La adaptación se produce cuando, inspeccionando uno o más aspectos de la entrega en desarrollo, se verifican las desviaciones en relación con los límites aceptables que pueden generar un producto que será rechazado.

Según [7], *Scrum* está formado por tres elementos principales: roles, procesos y artefactos. Entre los roles se destacan: el *Scrum Master*, el equipo del proyecto y el dueño del producto. Entre los procesos se encuentran: el inicio, la reunión de planificación del *Sprint*, el *Daily Scrum*, y la reunión de revisión de *Sprint*. Finalmente, en los artefactos se incluyen: el *Product Backlog*, *Sprint Backlog* y gráficas (*Burndown charts*) [12].

### 8.1.2 Kanban

*Kanban* es un método de organización y gestión del trabajo de servicios profesionales. Utiliza conceptos Lean como limitar el trabajo en progreso para mejorar los resultados. Un sistema *Kanban* es un medio para limitar el trabajo en curso y señalar cuando hay capacidad disponible para comenzar un nuevo trabajo. Esto se conoce como un "sistema de extracción" [13].

*Kanban* se basa en una estructura de flujo de trabajo continuo que mantiene a los equipos ágiles y listos para adaptarse a las prioridades cambiantes. Los elementos de trabajo, representados por tarjetas, se organizan en un tablero donde fluyen de una etapa del flujo de trabajo (columna) a la siguiente. Las etapas comunes del flujo de trabajo son tareas pendientes, en curso, en revisión, bloqueadas y lista [14].

### 8.1.3 Extreme Programming

La programación extrema (XP) es un marco de desarrollo de software ágil que tiene como objetivo producir software de mayor calidad y mayor calidad de vida para el equipo de desarrollo. *Extreme Programming* implementa un entorno simple pero efectivo que permite a los equipos ser altamente productivos. El equipo se organiza alrededor del problema para resolverlo de la manera más eficiente posible. XP está compuesta por cinco valores fundamentales para lograr un mejor resultado: comunicación, simplicidad, retroalimentación, respeto y valentía [15].

Los criterios fijados por el grupo de trabajo para la selección de la metodología de trabajo se describen a continuación:

1. Calidad del software: Que la metodología elegida ayude al desarrollo de un software de calidad.
2. Reuniones: Que comprenda reuniones con el equipo de trabajo para discutir la evolución del proyecto.
3. Flexibilidad: Que permita la adaptación más rápida a posibles modificaciones que se puedan presentar a lo largo del proyecto.
4. Monitoreo de las actividades: Que permita llevar el control de todas las actividades y de sus responsables.
5. Control de avance: Que permita saber en qué fase de avance se encuentra el proyecto en un momento dado.
6. Administración de requisitos: Que provea herramientas para la definición de los requisitos que componen el proyecto.
7. Documentación: La metodología debe proveer herramientas para una documentación clara y concisa del software a desarrollar.
8. Roles: Que la metodología posea una definición clara de los roles y las funciones para cada uno de ellos.

### 8.1.4 Metodología seleccionada

La metodología seleccionada por el grupo de trabajo el día 12 de febrero de 2019 fue *Scrum* **Ver Anexo A (Actas).** Esta decisión fue tomada debido a que *Scrum* se

ajusta a los criterios planteados por el grupo de trabajo. Por otro lado, se optó por realizar una reunión semanal de grupo donde se comunicarán avances, se hará retroalimentación a las tareas presentadas y se planificarán las siguientes actividades para cada uno de los miembros del grupo. Una imagen ilustrativa del modelo de ciclo de vida del proyecto se puede ver a continuación:

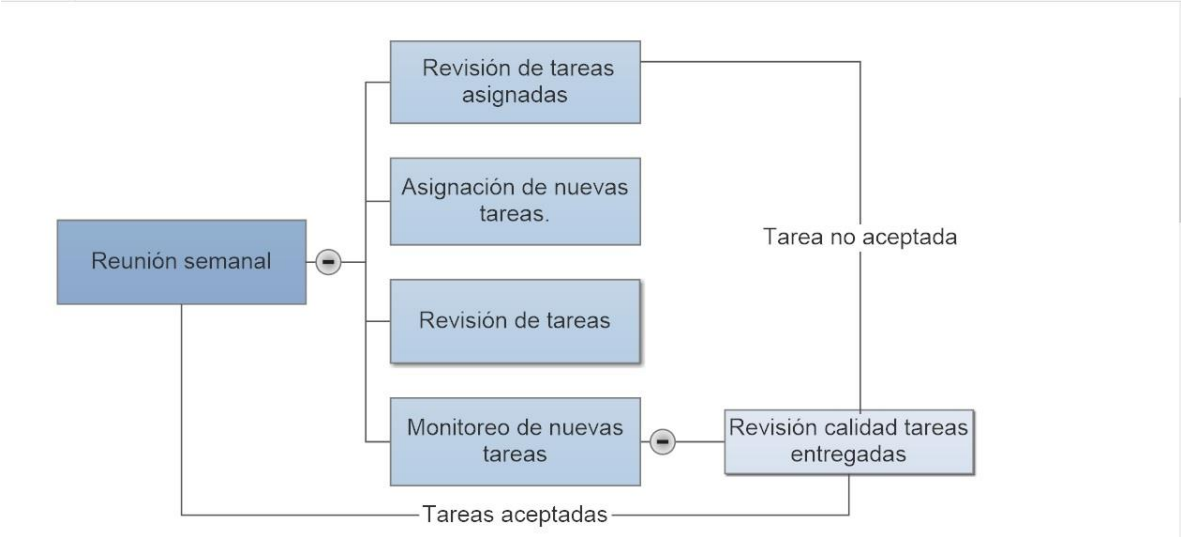


*Figura 1 Modelo ciclo de vida IngeSap*

Se especifica el estado de una tarea asignada: terminada, lista o retrasada. Una vez se haga la verificación de todas las tareas asignadas para una semana, se revisará si cumplen con los requisitos de calidad suministrados. Posteriormente, se empezará otro ciclo con nuevos objetivos a desarrollar por cada miembro del equipo. En caso de no estar lista se proseguirá a su revisión y posibles ediciones para terminarla.

## 8.2 Lenguajes y herramientas

A continuación, se presentarán los lenguajes y herramientas a usar durante el desarrollo del proyecto; éstas se seleccionaron de acuerdo con criterios que definió el grupo de trabajo.

### 8.2.1 Herramientas y lenguajes de desarrollo

Para la selección de lenguajes de desarrollo se definieron los siguientes criterios:

1. La tecnología debe ser multiplataforma: La tecnología seleccionada permite la ejecución del proyecto en diferentes sistemas operativos tales como *Android*, *iOS* y Web, sin necesidad de programar para cada sistema.
2. Curva de aprendizaje moderada: La tecnología seleccionada no tenga una sintaxis complicada. Además, que cuente con soporte, comunidad, librerías para la simplificación del trabajo y un manejador de paquetes.

3. Rendimiento: La tecnología seleccionada permite tener un buen rendimiento para el desarrollo de la aplicación.
4. Interfaz de usuario: La tecnología cuenta con componentes altamente personalizables y que se adapten de acuerdo con el dispositivo que se esté usando.

### 8.2.2 Desarrollo móvil

Para el desarrollo móvil se tuvieron en cuenta tanto *frameworks* y lenguajes. Luego de un respectivo análisis, el grupo de trabajo eligió las siguientes alternativas:

- **Ionic:** Es un marco de desarrollo HTML5 de código abierto para crear aplicaciones móviles híbridas; éstas utilizan el *WebView* de una plataforma móvil. La versión reciente de *Ionic* (versión 4) se construye utilizando Angular. Angular es una plataforma que facilita la creación de aplicaciones con la web; combina plantillas declarativas, inyección de dependencia, herramientas de extremo a extremo y mejores prácticas integradas para resolver los desafíos de desarrollo [16].

- **Xamarin:** Es una herramienta utilizada para el desarrollo de aplicaciones móviles multiplataforma que permite compartir aproximadamente el 90% del código en las principales plataformas. Al ser una herramienta relativamente nueva, se basa en la pila de tecnología de *Microsoft* y ya tiene una comunidad de más de 1.4 millones de desarrolladores [17].

La tecnología seleccionada fue Ionic, una tecnología multiplataforma, con una curva de aprendizaje relativamente corta, que permite hacer énfasis en la interfaz de usuario y un rendimiento muy bueno a la hora de correr aplicaciones. *Xamarin* también cumplía con todos estos criterios igualmente, solo que los miembros del equipo no contaban con experiencia trabajando en esta tecnología.

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

Para la selección de un manejador de bases de datos y el despliegue se fijaron los siguientes 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.

Bajo estos criterios, se consideraron dos tecnologías:

- **Firebase**: *Firebase* es un conjunto de herramientas para construir, mejorar y hacer crecer a las aplicaciones móviles, y las herramientas que le brinda cubren una gran parte de los servicios que los desarrolladores normalmente deberían construir. *Firebase* incluye cosas como análisis, autenticación,

bases de datos, configuración, almacenamiento de archivos, mensajería push, y otros servicios [18].

- **Azure:** *Azure* es una plataforma de nube flexible que permite crear, implementar y administrar rápidamente aplicaciones en una red global de centros de datos administrados por *Microsoft*. Permite: el despliegue de aplicaciones, autenticación, *hosting*, *machine learning*, almacenamiento entre otras [19].

Una vez estudiada cada una de las tecnologías se decidió usar *Firebase* por delante de *Azure*, esencialmente porque *Firebase* cumple con todos los criterios planteados: base de datos en tiempo real, provee un servicio de *hosting* para las aplicaciones y mecanismos de autenticación para los usuarios.

### 8.2.3 Herramienta 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 y se está convirtiendo rápidamente en el estándar para el control de versiones. *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. Se confirma el trabajo realizado localmente, y luego sincroniza su copia del repositorio con la copia en el servidor [20].

- **Subversión:** Es un sistema de control de versiones de software libre y de código abierto. Subversión maneja ficheros y directorios a través del tiempo. Hay un árbol de ficheros en un repositorio central [21].

Una vez planteadas las dos posibilidades el grupo de trabajo se inclinó a trabajar con *Git*, básicamente porque cumple a cabalidad con cada uno de los criterios planteados. *Git* es una herramienta muy completa para el control de versiones, los miembros del equipo han trabajado con ella, hay experiencia.

Acerca de la plataforma para el *hosting* de versiones, se consideraron las siguientes:

- **GitHub:** *GitHub* es un sitio web y un servicio basado en la nube que ayuda a los desarrolladores a almacenar y administrar su código por medio del control de versiones, así mismo ayuda a rastrear y controlar los cambios en su código [22].

Se eligió *GitHub* porque los integrantes del grupo poseen experiencia con la herramienta. Además, con el tutorial realizado en el aula de clase en el tema de administración de la configuración quedaron claros las posibilidades que se tiene con la aplicación escogida.

### 8.2.4 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:** *StarUML* es una herramienta de modelado de software de código abierto que admite *UML* (*Unified Modeling Language*). Se basa en la versión 1.4 de *UML*, proporciona once tipos diferentes de diagramas y acepta la notación *UML* 2.0 [23].

El equipo se inclinó por *StarUML* porque cumple a cabalidad con todos los criterios mencionados. Por otro lado, provee la gran mayoría de diagramas *UML* y permite colaboración entre miembros de un equipo de trabajo.

#### 8.2.4.1 Modelado de procesos de negocio

En cuanto al modelamiento BPMN las herramientas consideradas fueron:

- **Visual Paradigm:** *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 [24].
- **Bizagi Modeler:** *Bizagi Modeler*, también conocido con el nombre *Bizagi BPM Modeler* es una solución de *software* para el desarrollo de flujogramas y planificación de procesos para negocios que se rigen por las directrices del entorno BPMN 2.0 [25].

Una vez definidas estas dos alternativas se descartó *Visual Paradigm* porque dos miembros del equipo no están familiarizas con la herramienta. Con *Bizagi* todos los miembros del equipo tienen experiencia con la herramienta; además provee las herramientas para el correcto desarrollo del modelado de los procesos de negocio.

### 8.2.5 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 versionamiento de los documentos.

Las herramientas consideradas fueron las siguientes:

- **G Suite:** Es una plataforma basada en la tecnología de computación en la nube de *Google*. Provee correo electrónico, creación de documentos, hojas de cálculo, presentaciones, entre otros servicios para incrementar la productividad de los equipos de trabajo [26].

- **Microsoft Office 365:** Es un conjunto de productos ofimáticos orientados a mejorar la productividad en las compañías estos servicios están alojados en la nube y son provistos por *Microsoft*, algunas de las herramientas previstas por Office 365 son: Documentos, hojas de cálculo, presentaciones, correo electrónico entre otros servicios. Al estar alojado en la nube permite el trabajo en tiempo real y el versionamiento de los documentos [27]**.**

Se llegó a la conclusión de que se usarán las dos herramientas a lo largo del proyecto porque ambas cumplen con todos los criterios planteados.

### 8.2.6 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:** Trello es una herramienta de colaboración que organiza tus proyectos en tableros. Es decir, en Trello, se 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 [28].
  Trello usa tableros, listas, tarjetas para crear un sistema visual más simple de administración de proyectos y notificaciones sobre el estado cuando una tarea cambie de estado o se encuentre retrasada. Estos 4 elementos básicos

de la aplicación brindan un diseño visual de su trabajo para la toma de decisiones de alto nivel y un mejor control el proyecto [29].

- **TeamGantt:** *TeamGantt* es una herramienta web que permite organizar proyectos en forma de diagrama de Gantt. De sus principales características se encuentra el hecho de ser colaborativa. En general *TeamGantt* permite planificar el panorama general de un proyecto, para trabajar con un equipo de trabajo de forma intuitiva [30]. La única condición que no cumplía esta herramienta es la tercera, sin embargo, al ser tan intuitiva no fue tan importante tener experiencia en esta. Por otro lado, se escogió sobre herramientas como *Tom's Planner* porque ofrece muchos más servicios en su versión gratuita, además de ser un *power-up* de *Trello*. Esto significa que todas las tareas que se subieran a Trello automáticamente se enlazarían con TeamGantt.

- **Microsoft Planner:** *Microsoft Planner* es una forma de organizar el trabajo en equipo y las tareas para la gestión de tareas / proyectos. *Microsoft Planner* proporciona un centro para que los miembros del equipo creen planes, organicen y asignen tareas a diferentes usuarios y verifiquen las actualizaciones sobre el progreso a través de los paneles. También proporciona un lugar centralizado donde los archivos se pueden compartir y le da visibilidad a todo el equipo [31].

Una vez definidas las posibles herramientas se procedió a analizarlas, se observa que tanto Planner como Trello proveen servicios robustos para la planeación de proyectos e interfaces para la asignación, seguimiento de tareas y notificaciones sobre las mismas. Sin embargo, se seleccionó Trello por ser una herramienta de gestión de proyectos que nos ofrece gran cantidad de *power-ups*, que facilitan la integración con otras aplicaciones como *TeamGantt y Planning Poker* en comparación con otras herramientas. Es una gran ventaja a la hora de centralizar todos los procesos de administración y gestión del proyecto.

## 8.3 Plan de aceptación del producto

A continuación, se presentan los criterios para la aceptación del producto, que se acordaron entre el cliente y el equipo de *IngeSAP*, así como las técnicas y herramientas requeridas para medir el cumplimiento de los criterios establecidos.

### 8.3.1 Entregables

En la siguiente tabla se encuentran los entregables contemplados durante el avance y desarrollo del proyecto:

| **Entregable** | **Descripción** | **Dirigido a** | **Fecha límite de entrega** |
|---|---|---|---|

| | | | |
|---|---|---|---|
| SPMP | Documento que contiene la administración, planeación, vista general y contexto del proyecto. | Cliente | 27/02/2019 |
| SRS | Documento con la especificación de los requisitos del sistema | Cliente | 05/04/2019 |
| Primer Prototipo | Prototipo que contiene implementación del caso de uso más difícil. | Cliente | 05/04/2019 |
| SDD | Documento con la descripción del diseño del software | Cliente | Por definir |
| Segundo Prototipo | Prototipo que contiene al menos el 70% de los casos de uso implementados | Cliente | Por definir |

*Tabla 3 Entregables*

### 8.3.2 Criterios, Técnicas y Herramientas

Para que cada entregable sea aceptado se debe cumplir con los criterios de aceptación acordados previamente con el cliente. Adicionalmente, se consideraron las siguientes técnicas y herramientas que ayudarán a cumplir a cabalidad los criterios de aceptación de los clientes.

| 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.<br>Mendeley, para el manejo de las referencias bibliográficas. |
| SRS | 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.<br>Mendeley, para el manejo de las referencias bibliográficas. |
| SDD | Máximo 15 páginas, referencias IEEE, sin errores de ortografía, redacción clara y concisa, enumeración correcta | Microsoft Word, para la numeración, tabla de contenido, ortografía, |

| | | |
|---|---|---|
| | de tablas y figuras y tabla de contenido bien definida. | tablas y tablas.<br>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 sus respectivos criterios y herramientas*

## 8.4 Organización del proyecto y comunicación

Se enuncian las entidades externas o *stakeholders* que se encuentran asociadas al proyecto, así como el organigrama y la organización del equipo de trabajo de *IngeSAP*. Allí se hace énfasis en los roles que desempeña cada miembro del equipo, la descripción del rol pertinente, así como las responsabilidades asociadas a cada uno de ellos.

### 8.4.1 Interfaces externas o Stakeholders

En la tabla descrita a continuación, se presentan los *stakeholders*, personas interesadas e interfaces externas en la elaboración de *Healthy Routine*.

| **Nombre Entidad o Stakeholder** | **Descripción** | **Responsabilidad** | **Datos de contacto** |
|---|---|---|---|
| Anabel Montero | Stakeholder:<br>Profesora de catedra de la Pontificia Universidad Javeriana. Directora de la asignatura de Ingeniería de Software | -Encargada de proveer los lineamientos respecto a cada entrega del proyecto.<br>-Revisar y aprobar las entregas parciales del proyecto.<br>- Guiar el proyecto de software. | **Correo: anmontero @javeriana. edu.co**<br>**Teléfono: (+57) 310 283 4824** |

| | | | |
|---|---|---|---|
| Equipo de trabajo | Equipo de trabajo de la empresa IngeSAP | -Identificar los requisitos funcionales y no funcionales del aplicativo.<br>-Documentar el proceso de diseño e implementación de la aplicación.<br>-Presentar los avances del proyecto. | **Correo:** ororco_juan@javeriana.edu.co **Teléfono: (+57) 3225686346** |
| 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 |

*Tabla 5 Stakeholders*

### 8.4.2 Organigrama y descripción de los roles

Para el manejo eficaz del proyecto y de acuerdo a la metodología Scrum se asignó a cada miembro del equipo 1 roles (*Scrum Master, Product Owner* o *Scrum Team*), los cuales se definieron con base a el test de personalidad **Ver Anexo B(Test Hermann)** realizado por cada integrante del grupo y por los intereses expresados en cuanto a la afinidad con cada uno de estos roles.

#### 8.4.2.1 Organigrama

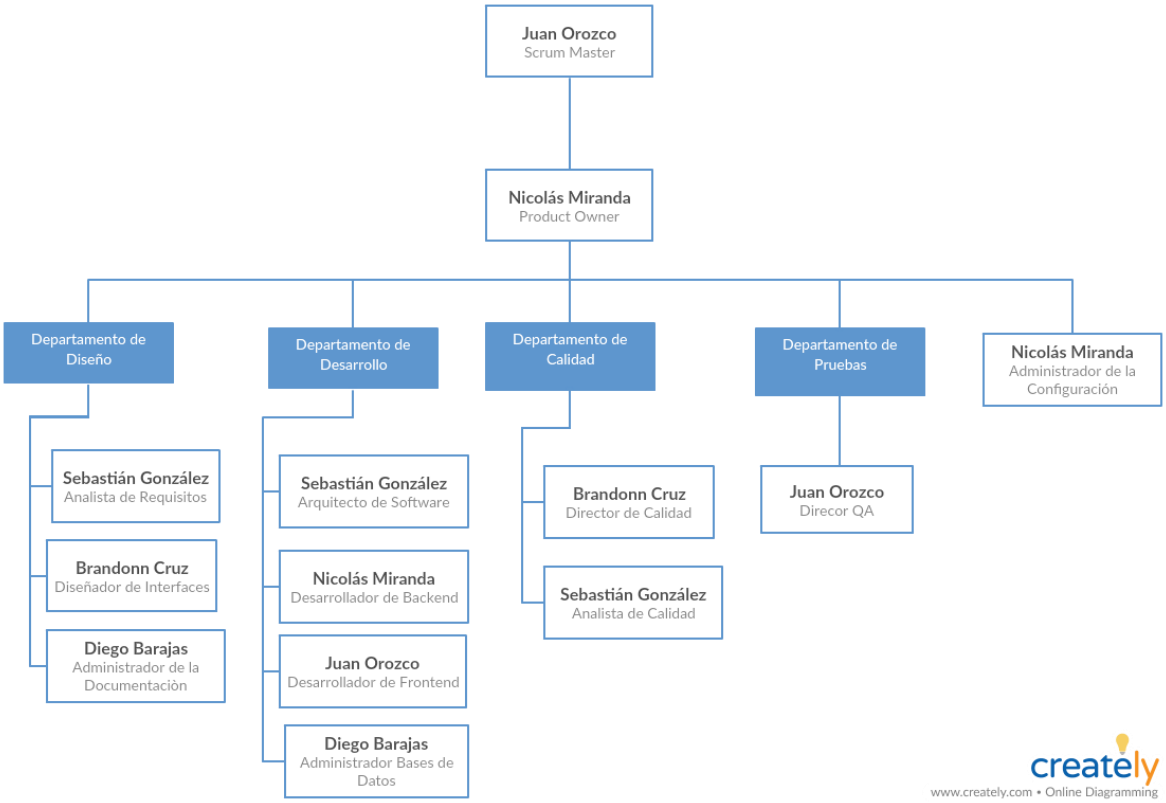


*Figura 2 Organigrama*

| **Rol** | **Descripción** | **Responsabilidades** |
|---|---|---|
| Scrum Máster | 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.<br>- Proteger al equipo de interrupciones y distracciones.<br>- Resolver conflictos internos.<br>- Hacer respetar las decisiones tomadas democráticamente.<br>- Asegurar la implementación de Scrum.<br>- Organizar reuniones del equipo. |

| | | |
|---|---|---|
| Desarrollador de Software | El equipo se conformará por miembros que se encarguen de manipular las tecnologías de desarrollo en cada una de las entregas y para cada Sprint a lo largo del proyecto. | - Elegir las tecnologías para el proyecto a lo largo del semestre.<br>- Desarrollar las funcionalidades con buenos estándares de programación, en lo posible el uso de patrones de diseño.<br>- Limitarse a la arquitectura diseñada desde un principio.<br>- Verificar que todos los componentes estén bien acoplados.<br>- Comunicar cualquier novedad al Scrum Máster. |
| 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.<br>- Verificar el desempeño de todos los miembros del equipo.<br>- Diseñar e implementar los índices de calidad.<br>- Encargado de revisar las recomendaciones planteadas por el docente.<br>- Diseñar planes para el manejo y control de riesgos. |
| Diseñador 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.<br>- Ilustración de ideas de diseño utilizando guiones gráficos, flujos de procesos y mapas de sitio.<br>- Diseñar elementos gráficos de la interfaz de usuario, como menús, pestañas y widgets. |

| | | |
|---|---|---|
| | | |
| Administrador de 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.<br>- Administrar la actividad de los datos.<br>- Administrar el Sistema Manejador de Base de Datos.<br>- Establecer el Diccionario de Datos.<br>- Asegurar la confiabilidad de la Base de Datos.<br>- Confirmar la seguridad de la Base de Datos. |
| Calidad y Control de Riesgos | 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.<br>- Verificar el desempeño de todos los miembros del equipo.<br>- Diseñar e implementar los índices de calidad.<br>- Encargados de revisar las recomendaciones planteadas por el docente.<br>- Diseñar planes para el manejo y control de riesgos. |

| | | |
|---|---|---|
| Administrador de configuración | 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.<br>- Creará los repositorios necesarios para gestionar el almacenamiento de la información.<br>- Encargado de llevar al tanto del directorio de versiones de los ítems de configuración. |
| Director de 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).<br>- Monitorear la documentación tanto del código como del diseño.<br>- Informar sobre la documentación al director de proyecto.<br>- Asociarse con los directores de desarrollo y configuración para generar los manuales de instalación. |
| Arquitectos | Encargados de planear la arquitectura tanto lógica como física del proyecto de acuerdo con los requisitos planteados. | - Verificar que todos los componentes estén bien acoplados.<br>- Diseñar los diagramas UML y la GUI (User Experience).<br>- Levantar requerimientos.<br>- Plantear la arquitectura del sistema. |

*Tabla 6 Descripción de roles*

# 9 Administración del proyecto

## 9.1 Métodos y herramientas de estimación

Para la estimación realizada en el presente documento, se consultó previamente la literatura referente al tema de administración de proyectos que engloba la estimación de recursos, costo y calendario.

Un proyecto de software está compuesto de múltiples piezas, por ende, es necesario aplicar técnicas de descomposición, bien sea del problema o del proceso [32]. En la estimación basada en el problema, se tienen en cuenta las métricas de productividad que son las líneas de código y los puntos de función, los cuales, sirven para dimensionar cada elemento de software [32]. También está la estimación basada en proceso, que se descompone en tareas más pequeñas y se estima el esfuerzo para terminar cada una [32].

Adicionalmente, a las técnicas de estimaciones anteriores, están los modelos de estimación empíricos [32] o modelos algorítmicos de costos [33], en los cuales se utiliza una fórmula matemática para poder predecir los costos del proyecto, en los cuales se tiene en cuenta el tamaño del proyecto y otros factores de software [33].

Las estimaciones descritas en los dos párrafos anteriores son las clásicas, sin embargo, en este proyecto se va a utilizar la metodología *Scrum* y, por lo tanto, técnicas de estimaciones ágiles. Así que, la técnica a implementar en el proyecto es *Planning Poker* y será descrita a continuación.

### 9.1.1 Planning Poker

Es una técnica de estimación bastante lúdica, pues se utiliza un mazo de cartas para realizar la planificación, pero en nuestro caso, se utilizó un *power-up* de *Trello* llamado *Planning Poker* para facilitar dicha tarea porque no se contaba con las tarjetas en físico y además facilita la centralización de la gestión del proyecto usando solamente *Trello*. Cada mazo contiene cartas y cada una tiene un valor numérico correspondiente a la serie Fibonnaci (0,1,2,3,5,8,13, 21) y algunas, tienen un signo de interrogación o infinito, como se muestra en la figura 3, que significa incertidumbre en cuanto a la estimación de la tarea [34]. Los valores anteriores, representan el número de puntos de la historia del usuario, el número de días ideales en los que se debe completar una tarea u otra unidad de estimación que el equipo proponga [35].


*Figura 3 Tarjetas SCRUM con la serie Fibonacci [Amazon]*

Con esta técnica de estimación, se asegura que todos los miembros del equipo compartan su opinión acerca del valor que debe tener cada tarea y que la reunión se vuelva más participativa e inclusiva [36].

El proceso de estimación mediante *Planning Poker* es el siguiente [35]:

1. Se empieza discutiendo acerca de las características de cada historia de usuario, los miembros del equipo pueden hacer preguntas si lo requieren, al *Product Owner,* quien no puede realizar estimaciones. Una vez se haya discutida la historia de usuario completa, cada participante selecciona una tarjeta de manera privada para realizar su estimación. Acto seguido, todos los participantes revelan los valores de sus estimaciones de manera simultánea.
2. Si todos los miembros del equipo tienen la misma valoración numérica de la estimación, ese es el valor que se va a convertir en la estimación, si no, los participantes están en todo el derecho de discutir acerca de su valoración, principalmente los que realizaron la estimación más alta y más baja. Después de esto, cada participante vuelve a seleccionar una tarjeta de estimación y de nuevo, se revela los valores de las estimaciones al mismo tiempo.
3. Este proceso se vuelve iterativo hasta que se logre un consenso entre todos los miembros del equipo o que la estimación de esa historia de usuario se debe aplazar porque se requiere información adicional al respecto.

Esta técnica de estimación ágil se realiza después de escribir el *Product Backlog* y durante todo el proyecto, es decir, es un proceso recurrente y participativo, por lo que se recomienda realizarlos justamente después de un *daily stand-up*, ya que todo el equipo está reunido [35].

## 9.2 Inicio del proyecto

La elección de roles se realizó mediante los resultados obtenidos en la prueba de Herrmann **ver anexo B (Test Hermann)**, que ayuda a conocer los estilos de aprendizaje, lo que deriva en las características y habilidades de cada individuo [37]. Por lo tanto, se propuso un plan de capacitación de acuerdo con el rol de cada miembro del equipo, para que adquiriese las habilidades necesarias para un óptimo desempeño de su cargo o, reforzar los conocimientos ya existentes.

### 9.2.1 Plan de capacitación

Se propone que el plan de capacitación sea totalmente autónomo y de manera autodidacta, siguiendo unos lineamientos planteados entre todos los miembros del equipo, para el cumplimiento de las actividades y tareas asignadas. A continuación, se van a describir por rol, las habilidades que se esperan adquirir o reforzar, la manera en cómo se va a aprender y cómo se realizará el proceso de aprendizaje.

- **Scrum Master (Juan Orozco):** Será el encargado de que se cumpla la filosofía de la metodología Scrum. Además, es el encargado de gestionar el control de reuniones con el equipo [38]. El *Scrum Master* debe saber en

profundidad todos los valores y las prácticas que implica la metodología ágil y todas sus características para el éxito del proyecto. Por lo tanto, Juan Orozco, de manera autodidacta va a realizar el curso gratuito *Applied Scrum for Project Management* disponible en la plataforma *edX* [39]. La idea es trabajar de 2 a 3 horas por semana para acabar todos los módulos en aproximadamente 1 mes.

- **Product Owner / Project Manager (Nicolás Miranda):** Es de vital importancia por sus habilidades blandas tales como la comunicación asertiva, el liderazgo [38] y la constante actualización sobre el mercado del *fitness*. De igual manera, se encarga de establecer los objetivos del producto y requisitos del sistema, monitoreando el progreso y que se cumpla el calendario de entregas [38]. Para que se puedan cumplir a cabalidad los objetivos de este rol, Nicolás Miranda, de manera autodidacta va a realizar el curso gratuito *Software Processes and Agile Practices* disponible en la plataforma *Coursera* [40]. La idea es trabajar de 2 a 4 horas por semana para acabar todos los módulos en aproximadamente 1 mes.
  El *Product Owner* debe entender muy bien la situación del mercado objetivo. Nicolás estará al tanto de rutinas de ejercicio y alimentación en las siguientes páginas web, que fueron escogidas minuciosamente, mediante los criterios establecidos por el grupo que fueron: calidad y rigurosidad en los artículos. Las páginas web son *PowerExplosive* [41] *Bodybuilding.com* [42] y *Muscle and Strength* [43].

- **Scrum Team (Sebastián González):** Sebastián se va a encargar en gran parte de la programación de la aplicación móvil, ya que es el mayor conocedor del framework *Ionic*, y para que se le facilite el desarrollo de la aplicación, va a ir revisando la documentación oficial de *Ionic* [44]. Por otro lado, estará encargado de la arquitectura de la aplicación, en particular, va a ayudar al diseño de los diagramas *UML*.

- **Scrum Team (Diego Barajas):** Al igual que Sebastián, Diego va a dedicar sus esfuerzos en la implementación del código de la aplicación y principalmente al manejo de la plataforma *Firebase* donde se tendrá alojada la base de datos en tiempo real y se utilizará el servicio de autenticación y alojamiento de *hosting*. Diego va a leer la documentación oficial de *Firebase* [45] para ir teniendo un primer acercamiento a la plataforma, y en el caso de tener problemas o dudas específicas, va a recurrir a la página *tutorialspoint* [46], en la cual se explica de una manera sencilla, los conceptos fundamentales de *Firebase* y hay algunos ejemplos que pueden servir de guía.

- **Scrum Team (Brandonn Cruz):** Se va a encargar del diseño de la interfaz de usuario móvil y la experiencia de usuario, ya que es una persona creativa y le gusta el diseño. Por lo que va a desarrollar un curso disponible en *YouTube* llamado *Mobile UI and UX Design Tutorial* [47], que fue el más sencillo y dinámico que encontró el equipo. También, se va a encargar de la administración de la configuración, es necesario que realice el curso *How to*

*Use Git and GitHub* [48]disponible en la plataforma *Udacity*, el cual, demora aproximadamente 3 semanas, si le dedica de 2 a 3 horas por semana, de manera autónoma.

### 9.2.2 Infraestructura del proyecto

Para iniciar el desarrollo del proyecto es necesario contar con herramientas tanto de hardware como de software que soporten el correcto desarrollo de las actividades. Por lo tanto, en las siguientes tablas se describirán: la infraestructura de hardware, relacionada con los equipos de cómputo que disponible cada integrante y las características técnicas que poseen (**ver tabla 7**). En la segunda tabla se describirán a groso modo las herramientas de software que se utilizarán en el proyecto (**ver tabla 8**).

| Dueño del equipo | Características técnicas |
|---|---|
| Juan Orozco | ● Lenovo<br>● Procesador Intel Core i5<br>● Memoria RAM de 6GB<br>● Sistema Operativo Windows 10. |
| Nicolás Miranda | ● MacBook Pro<br>● Procesador Intel Core i5<br>● Memoria RAM de 4GB<br>● Sistema Operativo macOS Mojave. |
| Sebastián González | ● Asus<br>● Procesador Intel Core i7<br>● Memoria RAM de 8GB<br>● Sistema Operativo Windows 10. |
| Diego Barajas | ● Hewlett-Packard<br>● Procesador AMD A8<br>● Memoria RAM de 8GB<br>● Sistema Operativo Windows 10. |
| Brandonn Cruz | ● Hewlett-Packard<br>● Procesador AMD A8<br>● Memoria RAM de 8GB<br>● Sistema Operativo Windows 10. |

*Tabla 7 Infraestructura de Hardware*

| Nombre | Descripción |
|---|---|

| herramienta | |
|---|---|
| Google Docs* | Es un procesador de texto, que permite crear, escribir, modificar documentos en línea, de manera gratuita y que, además, permite trabajar a la vez varias personas en un mismo documento [49]. |
| Mendeley Desktop | Es un manejador de referencias, que facilita la organización de referencias en documentos de texto y, además, es una red social académica, que permite compartir artículos de investigación y colaborar en ellos [50]. |
| SmartDraw | Es una herramienta para realizar diagramas, tales como diagramas de flujo, mapas mentales, organigramas, entre otros diagramas de negocios [51]. |
| TeamGantt | Es una herramienta online para realizar diagramas de Gantt, que ayuda a planificar la calendarización del proyecto [52]. |
| 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 [28]. |
| PlanningPoker | Es un juego de cartas digital que ayuda a los equipos que utilizan la metodología ágil Scrum, establecer efectivamente los objetivos de los sprint, a través de una planificación colaborativa y estimación basada en el consenso [53]. |
| Bizagi Modeler | Es una herramienta para el modelado de los procesos de negocio [25]. |
| 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 [54]. |
| GitHub | Es una plataforma de desarrollo colaborativo, que ofrece a los desarrolladores repositorios de software usando el sistema de control de versiones Git [22]. |
| Canva | Es una herramienta web para crear diseño y gráficos, tales como: logos, presentaciones, tarjetas, entre otros [55]. |

*Tabla 8 Herramientas de Software*

### 9.2.3 Tareas para realizar

Se usó *Trello* para establecer y asignar las tareas a cada persona de acuerdo con sus responsabilidad y rol. Del mismo modo se establecieron los *deadlines* para cada actividad y la frecuencia de estas. Para evidenciar el uso de esta herramienta **ver anexo C (*Trello*)**, donde se muestran ampliamente las funcionalidades que tiene Trello para la gestión del proyecto utilizando la metodología *Scrum*, tales como *Calendar*, *Planning Poker*, *TeamGantt* y *Burndown Chart*, mediante una serie de *screenshots*.

## 9.3 Planes de trabajo del proyecto

Es de suma importancia realizar planes de trabajo en cualquier proyecto que se emprenda, pues nos va a servir como hoja de ruta para alcanzar los objetivos planteados en la entrega.
Por lo tanto, se realizó un diagrama de descomposición de tareas (WBS) para describir las principales tareas a realizar en el proyecto. Después, se realizó un diagrama de Gantt, para tener claras las fechas de inicio y fin de cada actividad, incluyendo las dependencias entre ellas. Finalmente, se realizó un presupuesto del proyecto, mediante una tabla de flujo de caja, incluyendo entradas y salidas de dinero de las actividades.

### 9.3.1 Estructura de descomposición de tareas (WBS)

Como dice [56] el WBS, sirve para representar todo el trabajo que se debe realizar en un proyecto, nos dice lo que hay que hacer y, además, establece los límites. Para esta entrega, se tomó en cuenta las secciones más relevantes, propuestas en el documento SPMP y se realizó el diagrama WBS **ver anexo D (Diagrama WBS)**.

### 9.3.2 Calendarización del proyecto

El diagrama de Gantt es una herramienta para ayudar a los gerentes de proyectos a planificar las tareas y programar las mismas, durante un periodo de tiempo determinando dia [57]. Ayuda a gestionar el uso de recursos requeridos, tanto humanos como materiales. Además de lo anterior, nos ayuda a identificar dependencias entre las tareas, pudiendo identificar puntos críticos y así, poder tomar un plan de acción para mitigar el problema [57].

Para la presente entrega se realizó un diagrama de Gantt utilizando la herramienta *Trello* y *TeamGantt* **ver anexo E (Diagrama Gantt)**, en el cual se pueden evidenciar las fechas de inicio y finalización de cada actividad y las relaciones de precedencia entre algunas actividades que la tenían. Se tuvo en cuenta la metodología *Scrum* y se agruparon en *Sprints* las tareas, que se asignaron a cada integrante del equipo, de acuerdo con sus habilidades y conocimiento. En total, el cronograma tiene 3 *Sprints*, uno por semana.

### 9.3.3 Presupuesto del proyecto

De acuerdo con la metodología seleccionada (**ver sección 8.1.4**) se decide estimar los costos del proyecto de acuerdo con las horas trabajas por los miembros en cada sprint (3 Sprints). Es importante mencionar que según la metodología *Scrum* existen 3 roles o cargos principales que son: *Scrum Master*, *Product Owner* y el *Scrum Team*. Cabe aclara que el *Scrum Team* se dividió en diferentes cargos los cuales fueron: arquitecto de software, desarrollador *backend* y *frontend*, administrador de bases de datos, analista de calidad, diseñador de interfaces, líderes de pruebas y administración de la configuración.

Se consultaron los salarios del Scrum Master y Product Owner, en diferentes páginas de Colombia y no se encontró datos relevantes, entonces, se decidió consultar el *Estudio de Salarios y Profesionales del Sector de Software y TI en Colombia 2017* [58], para hacer una equivalencia, de acuerdo a las responsabilidades y tareas de cada uno. También se consultó el salario promedio de un diseñador de interfaces, en páginas como *opcionempleo.com* [59], *elempleo.com* [60] y *computrabajo.com* [61] ya que este cargo no estaba contemplado en el estudio anteriormente mencionado.

Para calcular el salario por horas de trabajo, de cada integrante del equipo **ver anexo F (Presupuesto),** se tomó en cuenta que en Colombia la jornada laboral aumentó de 8 a 10 horas, como lo informa [62]. Por lo tanto, al mes se trabajan aproximadamente 205 horas, según [62].

Finalmente se estimó el costo del proyecto teniendo en cuenta el tiempo invertido en los Sprints por cada uno de los miembros y en cada uno de los roles que ellos tienen **ver anexo F(Presupuesto).** Por otra parte se tienen en cuenta imprevistos.

# 10 Monitoreo y control del proyecto

En este apartado se describen las herramientas y técnicas que se usarán para monitorear el desarrollo del proyecto. Principalmente, con el fin de que los administradores del proyecto tomen las decisiones adecuadas teniendo en cuenta los tiempos establecidos. Además, se describirán los protocolos que se seguirán en caso de que se presente algún problema.

## 10.1 Administración de requisitos

El proceso que se utilizará para detectar, reportar y controlar los cambios que va a sufrir los requisitos del producto es:
Cuando el *Product Owner* considere necesario realizar cambios en los requisitos deberá informar al *Scrum Team*, quienes deberán estudiar la forma en que estos cambios afectarán al proyecto, tanto en la modificación de artefactos de planeación,

como en partes del sistema ya implementadas. El *Scrum Team* también determinará si estos cambios son posibles. Si no lo son, especificar las razones por las que no son viables los cambios y cuáles modificaciones se realizarían para que sean viables.

Después de que el *Scrum Team* ha terminado el análisis de la propuesta, deberá enviar un reporte al *Product Owner*, quien revisará si continúa aceptando las consecuencias y los cambios que el *Scrum Team* ha propuesto. Por otro lado, si decide realizar cambios se enviará una nueva solicitud.

A continuación, se muestra el diagrama BPMN de este proceso:

**Ver anexo G (Diagramas BPMN)**

## 10.2 Monitoreo y control del progreso

Aquí se define la rúbrica que se utilizará para medir el progreso del proyecto. Adicionalmente se describirá la forma en que los integrantes del grupo reportarán el trabajo realizado.

### 10.2.1 Medidas del proyecto

El proyecto se define como una serie de funcionalidades que en conjunto forman al sistema, el producto final. No obstante, una condición irrevocable del proyecto es la fecha de entrega, es por este motivo que la descripción de unidades del proyecto estará basada en una *calendarización.* "La calendarización del proyecto de software es una acción que distribuye el esfuerzo estimado a través de la duración planificada del proyecto, asignando el esfuerzo a tareas específicas de ingeniería de software" [32].

"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" [32].

Debido a que este proyecto está conformado por: conjunto de tareas, fecha límite de entrega y un grupo de trabajo; conformando así las principales entradas de un *cronograma* o *gráfico de Gantt* [32]; el progreso de este proyecto está determinado por el número de tareas terminadas en el tiempo establecido; es decir, este proyecto avanzará a medida que las tareas del gráfico de Gantt sean completadas.

**Medida:** Tareas terminadas / día

### 10.2.2 Actividades para realizar

Para el control de actividades se utilizará la herramienta web *TeamGantt.* Ésta permite en su versión gratuita incluir a tres personas y crear un proyecto. Se seleccionó principalmente por su excelente calidad y por su interfaz intuitiva, ya que permite trabajar velozmente. En este diagrama de Gantt se describe las actividades

que se deben realizar, las fechas de inicio y cierre, los tiempos establecidos y los responsables de completarlas.

Según [32]. hay distintas formas de hacer seguimiento a un calendario; sin embargo, debido a la magnitud de este proyecto que es más bien pequeña, se decidió escoger sólo las siguientes recomendaciones:

- "Realizar reuniones periódicas del estado del proyecto, en las que cada miembro del equipo reporte avances y problemas." [32] Por convenio el grupo designó los martes para realizar estas reuniones periódicas, con una duración de dos horas. Además, si se presentas inconvenientes, el *Scrum Master* puede convocar reuniones de emergencia que se llevarán a cabo los días lunes.
- "Comparar la fecha de inicio real con la fecha de inicio planeada para cada tarea de proyecto mencionada en la tabla de recursos" [32]. Con el fin de revisar si el responsable de la tarea ha incurrido en una falta, para así tomar las acciones correctivas necesarias.
- "Reunirse informalmente con los profesionales para obtener su valoración subjetiva del avance a la fecha y los problemas en el horizonte" [32]. Esta técnica está más enfocada hacia el docente del curso de Ingeniería de Software, pues el grupo se pondrá en contacto con éste cuando lo considere necesario.

### 10.2.3 Acciones correctivas

La siguiente tabla presenta las acciones que se tomarán cuando alguno de los miembros del grupo incumpla las reglas. Estas sanciones están en forma de puntuación. El grupo de trabajo considera que la mejor forma de evitar las faltas por parte de los miembros es mediante un sistema de calificación en el que todos los integrantes del grupo inician con una nota de 5.0, si esta disminuye a 3.5 el grupo completo debe hablar con el miembro que ha cometido faltas. Si la nota baja hasta 2.5, el administrador del curso procederá a comunicarle esta situación a la docente encargada.

| **Falta** | **Descripción** | **Sanción** |
|---|---|---|
| Actividad no iniciada en el tiempo establecido. | El responsable de la actividad no ha comenzado la actividad en el momento de su entrega. | - 1.0 |
| Actividad parcialmente realizada. | El responsable ha comenzado la actividad, pero su trabajo es muy incompleto en comparación con lo esperado. | - 0.5 |

| | | |
|---|---|---|
| Actividad parcialmente terminada. | Al momento de presentar la actividad, el responsable muestra resultados casi satisfactorios. | - 0.3 |

*Tabla 9 Sanciones*

Lo dicho anteriormente permite establecer una ruta de acción al momento de imponer sanciones a alguno de los miembros del grupo; ahora se describirán las técnicas que se utilizarán para continuar con el proyecto luego de que se dé uno de estos inconvenientes:

- **Time-boxing:** Esta estrategia se utiliza cuando se asume que el producto no podrá ser entregado a tiempo, entonces se elige un paradigma incremental [32]. Es decir que, si en el proyecto se presentan atrasos con las tareas, se pasará a darles un tiempo extra donde se implemente las partes más esenciales de estas, posponiendo lo demás que podría ser prescindible.

## 10.3 Cierre del proyecto

Para el cierre del proyecto se utilizará un proceso *de Postmortem,* éste es implementado para mejorar a partir de las experiencias del proyecto; es decir, un mejoramiento continuo del proceso [63]. El objetivo que se busca con el *Postmortem* es evaluar el desempeño del equipo de trabajo, para mejorar en proyectos futuros [63].

**Criterios de entrada:**

El producto ha sido terminado y probado. El equipo posee datos que evidencian su desempeño durante el desarrollo del proyecto.

**Revisión de los datos del proceso:**

- Examinar los datos del trabajo realizado por el grupo y por cada individuo.
- Identificar en dónde hubo fallas.
- Comparar el desempeño realizado con lo planeado al comienzo del proyecto.

Esta revisión dará como resultado los problemas que surgieron, para posteriormente plantear como afrontarlos [63].

**Revisión de los datos de calidad:**

- Comparar los datos de calidad con los planeados.
- Evaluar la calidad del producto.

Aquí se busca identificar si los criterios usados para la calidad podrían mejorarse [63].

**Revisión de los datos de calidad:**

- Comparar con *Postmortems* pasados.
- Preparar el PIP (*Personal Investment Performance*)

Revisar si las mejoras propuestas por *Postmortems* pasado fueron aplicadas, y si además estas mejoraron el proceso de trabajo. Además, se quiere saber si son necesarias nuevas modificaciones [63].

**Evaluación de roles:**

- Evaluar el trabajo realizado por los roles establecidos.

Aquí se va a evaluar el desempeño de cada uno de los roles en el proyecto, revisando el trabajo que éste realizó, los recursos que utilizó, los problemas que tuvo, los aspectos en que podría mejorar y cómo lo va a hacer. Esta actividad es responsabilidad del líder del proyecto (en este proyecto el *Scrum Master*) [63].

**Reporte del ciclo:**

- Describir los resultados, el proceso que se usó y los roles implicados.
- Informar dónde hubo o no fallas.
- Explicar el desempeño de cada miembro del grupo, tanto en su rol individual como en su rol de desarrollador.

Este apartado debe ser breve pero claro con lo aprendido. Sirve como conclusiones, las cuales deben estar justificadas con los datos generados en el desarrollo del proyecto [63].

# 11 Entrega del producto

## 11.1 Entrega del producto a futuro

El producto se distribuirá entre los usuarios por medio de un url con acceso público a los archivos en una plataforma de Dropbox, en este link se encontrarán tanto la aplicación *Healthy Routine* en forma de archivo con extensión (.apk), como también un pequeño e intuitivo manual de usuario para facilitar la integración de la app en la vida cotidiana de nuestros usuarios.

## 11.2 Compatibilidad

El producto en primera instancia sólo funcionará para sistemas operativos Android con sistema operativo *Android Ice Cream Sandwich* (4.0) o dispositivos con sistema operativo iOS.

# 12 Procesos de soporte

En este apartado se muestran los procesos transversales al proyecto, los cuales, funcionan como valor agregado y que permiten la correcta ejecución de este.

## 12.1 Ambiente de trabajo

Para generar un ambiente de trabajo ideal dentro del grupo, IngeSap estableció un conjunto de reglas **Ver Anexo A (Actas)**, con el fin de que exista una buena convivencia dentro del grupo, 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 las reglas y se procederá a colocar una penitencia, si es necesario. Las reglas establecidas se describen a continuación.

### 12.1.1 Reglas de trabajo

1. Cada miembro del equipo debe dirigirse a los demás con respeto.
2. Cada miembro del equipo debe ser responsable por las tareas que le fueron asignadas cada semana.
3. Los problemas personales del equipo deben ser resueltos sin que perjudiquen el avance del proyecto.
4. El principal medio de comunicación será el grupo de *WhatsApp*.
5. En las reuniones semanales estará prohibido tocar temas que no estén relacionados con el proyecto.
6. 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.
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 un 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.

### 12.1.2 Excusas válidas

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

1. Emergencia médica.
2. Cita médica programada.
3. Razones personales como una tragedia familiar.
4. Eventos que el integrante pueda tener tales como actividades universitarias o salidas de campo.
5. Enfermedad que no le permita la asistencia y normal participación dentro de la reunión.

### 12.1.3 Sanciones y acciones correctivas

Si uno de los miembros de *IngeSAP* 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 la entrega de comida para compartir con el grupo hasta la expulsión del integrante del equipo de trabajo.

## 12.2 Análisis y administración de riesgos

Esta sección presenta el análisis y administración de riesgos que desarrolló IngeSAP, IngeSAP reconoce la importancia de una clara gestión de los riesgos en un proyecto, por lo que también se describen los procesos para cada uno de los riesgos considerados para el proyecto.

### 12.2.1 Análisis de riesgos

Se define como riesgo toda posibilidad de ocurrencia de aquella situación que pueda entorpecer el normal desarrollo de las funciones y actividades de una empresa que impidan el logro de sus objetivos, en cumplimiento de su misión y su visión. Se refiere a la variabilidad de los beneficios esperados por los inversionistas [64].

Después de identificar y clasificar los riesgos se continúa a realizar el análisis de estos, es decir que se estudia la posibilidad y las consecuencias de cada factor de amenaza con el fin de establecer la vulnerabilidad del proyecto. El análisis determinará cuáles son los factores de riesgo que potencialmente tendrían un mayor efecto sobre nuestro proyecto [65].

De acuerdo con el PMBOK [66], el análisis y administración de riesgos se encuentra divido en cuatro etapas. Estas cuatro etapas incluyen desde la definición de los riesgos antes de iniciar el proyecto, monitorización de cada uno, acciones a realizar ante su presencia y la modificación a lo largo del proyecto.

Principalmente se realiza la identificación y clasificación de los riesgos que se puedan presentar en el proyecto, luego se procede a describir aquellos que se presentaron, el siguiente paso es cuantificarlos; va de la mano con la probabilidad de que este riesgo ocurra y de su posible impacto en el proyecto. La siguiente ecuación describe la relación entre la cuantificación del riesgo, su impacto y probabilidad.

$$Cuantificación\ riesgo = probabilidad\ de\ ocurrencia * impacto\ sobre\ el\ proyecyo$$

**(1) Ecuación para cuantificar un riesgo**

Una vez realizado este proceso, se establecen los planes de mitigación y acción contra cada uno de los riesgos establecidos. Finalmente, se debe realizar un control y monitoreo constante sobre cada uno durante el desarrollo del proyecto.
IngeSAP realizó todo el proceso correspondiente al análisis y gestión de riesgos, este documento puede ser consultado en **Ver Anexo H (Riesgos.xls).**

### 12.2.2 Clasificación de riesgos

Para clasificar los riesgos se utilizaron los riesgos propuestos por [65]. En esta clasificación se encuentran los riesgos más comunes que pueden ocurrir en un proyecto y se realiza una definición clara de cada uno de los riesgos. Las clasificaciones se encuentran en la siguiente tabla.

| Clasificación | Descripción |
|---|---|
| Riesgo de presupuesto | Una mala estimación del proyecto puede llevar a afectar la financiación del proyecto en su transcurso. |
| Riesgos de horario | Riesgo que afecta la programación del proyecto y puede llevar a posibles retrasos en la entrega. |
| Riesgos operacionales | Riesgo de pérdida debido a la falla en la implementación de los procesos del sistema o algunos riesgos de eventos externos. |
| Riesgos técnicos | Los riesgos técnicos generalmente conducen a fallas de funcionalidad y rendimiento. |
| Riesgos programáticos | Estos son los riesgos externos más allá de los límites operacionales de la empresa. Todos estos son riesgos inciertos y que están fuera del control del equipo de trabajo. |

*Tabla 10 Clasificación y descripción de los riesgos*

### 12.2.3 Cuantificación de los riesgos

Cada riesgo identificado por IngeSAP se le asignará dos métricas. La primera indicará la probabilidad de ocurrencia del riesgo en el proyecto, mientras que la segunda indicará el impacto que tendría dicho riesgo si ocurriese sobre el proyecto. La tabla para la cuantificación de dichas métricas se presenta a continuación.

| Cuantificación | Probabilidad | Descripción |
|---|---|---|
| 0 - 0.19 | Muy baja | El riesgo no se presenta durante el desarrollo del proyecto. |
| 0.20 - 0.39 | Baja | El riesgo se presenta en pequeñas etapas durante el desarrollo del proyecto. |
| 0.40 - 0.59 | Media | El riesgo puede ocurrir en la en varias circunstancias durante el desarrollo del proyecto. |
| 0.60 - 0.79 | Alta | Se espera que el riesgo esté presente en casi todas las etapas durante el desarrollo del proyecto. |
| 0.80 - 0.99 | Muy Alta | Se espera que el riesgo esté presente en todas las etapas del proyecto. |

*Tabla 11 Probabilidad de ocurrencia de un riesgo*

| Cuantificación | Impacto | Descripción |
|---|---|---|
| 0 - 0.19 | Despreciable | El riesgo no tendrá impacto en el proyecto. |
| 0.20 - 0.39 | Menor | El riesgo tendrá un bajo impacto en el proyecto. |
| 0.40 - 0.59 | Moderado | Este riesgo tendrá un impacto mediano sobre el desarrollo del proyecto |
| 0.60 - 0.79 | Mayor | Si el riesgo ocurre tendrá un efecto muy alto sobre la ejecución del proyecto. |
| 0.80 - 0.99 | Catastrófico | Si el riesgo ocurre tendrá un efecto muy grave sobre la ejecución del proyecto. |

*Tabla 12 Impacto de ocurrencia de un riesgo*

## 12.3 Administración de configuración y documentación

A continuación, se muestran los ítems de configuración (diferentes versiones) que han sido identificados para el desarrollo de este proyecto, en base a lo visto en el cursode Ingeniería de Software:

| Ítem de configuración | Descripción | Momento de creación y modificación |
|---|---|---|
| WBS (*Work Breakdown Structure*) | Estructura jerárquica que permite la descomposición del trabajo que se debe realizar. Los niveles inferiores representan detalle de los superiores [56]. | Creado en la primera entrega del proyecto, pudiendo ser modificado únicamente por todo el grupo en las reuniones semanales. |
| SPMP (Software Project Management Plan) | Documento formal que contiene la planificación del proyecto y todo lo que esto conlleva, como los entregables del proyecto, metodologías, fechas de entregas, etc. En esencia permite tener una visión general del proyecto [67]. | Puede ser modificado si el Scrum decide hacer cambios en la planificación o si es necesario realizar correcciones solicitadas por el docente del curso. |
| Product Backlog | Listado de los requisitos del sistema. Se usará durante todo el ciclo de vida, y puede estar sujeto a cambios para incorporar las necesidades del sistema o del *Product Owner*. | Creado al inicio del ciclo de vida. Puede ser modificado si el *Scrum Team* o el *Product Owner* así lo consideran, sin embargo, es sujeto a las restricciones de tiempo que un cambio en la funcionalidad conlleva. |
| Diagrama de casos de uso | Representación de las funcionalidades del sistema. | Se crea al finalizar el SPMP. Es creado y modificado por el *Scrum Team*. |
| Diagrama de clases | Representación estática de los objetos del problema. | Se crea al finalizar el SPMP. Es creado y modificado por el *Scrum Team*. |

| | | |
|---|---|---|
| Diagrama de Gantt | Diagrama que sirve a modo de cronograma para el trabajo que se debe realizar [57]. | Creado desde la primera entrega (**ver anexos**). Puede ser modificado por el encargado del monitoreo en este proyecto cuando se realizan las reuniones semanales. |
| Código fuente | Archivos en los cuales se está escribiendo el código del producto. | Creado luego de haberse realizado la primera entrega. Todos los integrantes pueden modificarlo en cualquier momento (siguiendo las pautas de la administración de la configuración). |
| Tabla riesgos | Tabla con los riesgos identificados, con la su probabilidad de ocurrencia e impacto. | Creada desde la primera entrega (**véase la sección 12.2 de este documento**). |

*Tabla 13 Ítems de configuración*

Cabe aclarar que el proceso de control de cambios a los ítems de configuración incluye tres tipos:

1. Relacionado a las funcionalidades del producto; son aquellos que tienen que ver tanto con cambios en los requisitos como con la reparación de errores. Es decir, que afectarían tanto a artefactos de diseño como a código fuente del producto.
2. Relacionado a la documentación; son aquellos artefactos que se utilizarán en todo el ciclo de vida y que pueden ser modificados, por ejemplo, la tabla de riesgos o el diagrama de Gantt utilizado para el control de actividades.

El proceso comienza cuando el *Scrum Team* ha identificado un cambio, por lo tanto, debe preguntarse a cuál de los dos cambios anteriormente mencionados pertenece. Si el cambio pertenece a funcionalidad debe realizarse el proceso *Modificar requisitos* descrito en la sección 10.2 de este documento. Además, se debe llenar un formato CRF al mismo tiempo, luego debe redactarse un breve resumen del cambio en el acta de la última reunión realizada.

Si el cambio pertenece a la documentación, este debe ser añadido a la tabla de cambios y registrado en las actas de la última reunión realizada.

**Ver anexo G (Diagramas BPMN: Proceso de control de cambios)**

Por último, cabe aclarar que 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 de esto se debe llevar registro en el acta, ya dependerá del equipo de trabajo si los cambios permanecen o no, para ser llevados a las herramientas de control de versiones mencionadas en la sección 8.2.3.

## 12.4 Control de calidad

**Medidas de Calidad:**

Debido a que hay múltiples características que se deberían cumplir para entregar un proyecto con calidad, tanto en el ámbito de la documentación como en el de desarrollo de Software, se ha decidido desarrollar una estrategia que permita flexibilidad al momento de determinar los factores de la calidad para una determinada tarea. Esto aprovechando el hecho de que se trabajará con una metodología ágil como Scrum que permite hacer revisiones constantemente sin afectar el rendimiento del proyecto demasiado.

Como medida que se tomará para tener un marco de referencia en cuanto a factores de calidad, se utilizará con respecto a Software los factores de calidad de la *ISO 9126*, que son: funcionalidad, confiabilidad, usabilidad, eficiencia, facilidad de recibir mantenimiento y portabilidad [32]. En la parte de documentación, el principal marco de referencia será el cumplir con las características de presentación exigidas por el docente del curso de Ingeniería de Software.

Se puede decir entonces que la calidad de cada una de las tareas se verá reflejada en el número de características que el equipo identificó, con las que fueron implementadas exitosamente. Por ejemplo, si se pide un diagrama BPMN del proceso para control de calidad, las características que podría pedir el docente es que sea claro, y las características que podría identificar el **Scrum Team** es que sea desarrollado en Bizagi; luego la calidad del modelo BPMN será la máxima si es implementado con estas dos características.

**Proceso para control de calidad de la documentación:**

El proceso comienza en una de las reuniones del *Scrum Team*, al momento de terminar la definición de las tareas de cada Sprint (desarrollo del Sprint Backlog) el equipo de trabajo comenzará a desarrollar un plan de gestión de calidad, en el cual definirá las características que debe tener la documentación que se realizará en cada Sprint. Aquí se generará una hoja de verificación con estas características, que posteriormente se diligenciará.

Cuando el Sprint se ha terminado en la reunión semanal del *Scrum Team*, es necesario diligenciar la hoja de verificación. Si las tareas no fueron terminadas, debe

documentarse la razón con el fin de que la tarea sea puesta para el siguiente Sprint y que los problemas sean resueltos.

**Ver anexo G (Diagramas BPMN: Controlarla calidad de la documentación)**

**Proceso para el control de calidad del software:**

Durante la reunión en la que el *Scrum Team* ha terminado de redactar el Sprint Backlog, el equipo debe redactar una hoja de verificación con la definición de las tareas y las características que deben ser completadas.

En la siguiente reunión semanal al terminar el Sprint, cada miembro del equipo debe diligenciar la hoja de verificación. Si el responsable de una determinada tarea no la terminó, en las conclusiones debe explicar el por qué, para que la tarea sea resuelta en el siguiente Sprint.

Si la tarea fue hecha y el responsable demuestra que se ha realizado pruebas de su funcionamiento, el codeline se añadirá al baseline.

**Ver anexo G (Diagramas BPMN: Controlar la calidad del software)**

# 13 Anexos

A. Actas (Acceso en https://bit.ly/2EG12Ng)
    a. Acta 1 (Acceso en https://bit.ly/2VvPgdU)
    b. Acta 2 (Acceso en https://bit.ly/2GUI6wz)
    c. Acta 3 (Acceso en https://bit.ly/2VvPTnM)
    d. Acta 4 (Acceso en https://bit.ly/2T64uJM)
    e. Acta 5 (Acceso en https://bit.ly/2tNOjBG)
B. Test Hermann (Acceso en https://bit.ly/2HcpsiT)
C. Trello (Acceso en https://bit.ly/2VsmI4V)
D. Diagrama WBS (Acceso en https://bit.ly/2UfVKNQ)
E. Diagrama Gantt (Acceso en https://bit.ly/2Tq9LLE)
F. Presupuesto (Acceso en https://bit.ly/2Ti2Nsf)
G. Diagrama BPMN (Acceso en https://bit.ly/2Caclf9)
H. Riesgos (Acceso en https://bit.ly/2IKSMiU)
I. Hojas de verificación (Acceso en https://bit.ly/2ErXs80)
J. Reporte gerencial (Acceso en https://bit.ly/2Vv9rIN)
K. Lista de requisitos (Acceso en https://bit.ly/2EGPjxU)

# 14 Referencias


[1] H. Line, “What to keep in mind,” 2016. [Online]. Available: https://www.healthline.com/health/mental-benefits-sports#keep-in-mind.

[2] Significados, “Significado de Alimentación,” 2019. [Online]. Available: https://www.significados.com/alimentacion/.

[3] V. Comunicaciones, “Definición de Caloría,” 2018. [Online]. Available: https://conceptodefinicion.de/caloria/.

[4] A. Marcet, “Las 100 palabras clave del fitness,” 2014. [Online]. Available: https://www.sportlife.es/entrenar/fitness/articulo/100-palabras-fitness.

[5] Definicion.de, “Definición de Dieta,” 2016. [Online]. Available: https://definicion.de/dieta/.

[6] Cuidate Plus, “Fitness.” [Online]. Available: https://cuidateplus.marca.com/ejercicio-fisico/diccionario/fitness.html.

[7] Vitalia, “Qué es Nutrición,” 2016. [Online]. Available: https://vitalia.es/b13m93/nutricion-que-es.

[8] C. Plus, “Proteínas,” 2018. [Online]. Available: https://cuidateplus.marca.com/alimentacion/diccionario/proteinas.html.

[9] F. GmbH, “Pushups: Mucho más que un entrenamiento para los pectorales,” 2019. [Online]. Available: https://www.freeletics.com/es/blog/posts/ejercicios-de-freeletics-push-ups/.

[10] “Rutina,” *Gran Diccionario de la Lengua Española*, 2016. [Online]. Available: https://es.thefreedictionary.com/rutina.

[11] F. Training, “Stretching. ¿Qué es? ¿Cómo hacerlo? Beneficios,” 2015. [Online]. Available: http://vitalstrechbalance.over-blog.com/2015/02/stretching-que-es-como-hacerlo-beneficios.htmlz.

[12] J. S. Ken Schwaber, “The Scrum Guide™,” 2018. [Online]. Available: https://www.scrumguides.org/scrum-guide.html.

[13] D. Radigan, “Kanban.” .

[14] M. Rehkopf, “Kanban vs. Scrum.” .

[15] D. Wells, “Extreme Programming: A gentle introduction,” 2013. [Online]. Available: http://www.extremeprogramming.org/.

[16] Techcronus, “Benefits IONIC Framework App Development,” 2018. [Online]. Available: https://www.techcronus.com/blog/benefits-ionic-framework-app-development/.

[17] Altexsoft, “The Good and The Bad of Xamarin Mobile Development,” 2018. [Online]. Available: https://www.altexsoft.com/blog/mobile/pros-and-cons-of-xamarin-vs-native/.

[18] C. Esplin, “What is Firebase?,” 2016. [Online]. Available: https://howtofirebase.com/what-is-firebase-fcb8614ba442?gi=ff22775efbdb.

[19] Microsoft, “¿Qué es Azure?,” 2018. [Online]. Available: https://azure.microsoft.com/es-es/overview/what-is-azure/.

[20] Kayla Ngan, “What is Git?,” *Microsoft*, 2018. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/learn/git/what-is-git.

[21] B. W. F. & C. M. Pilato, “¿Qué es Subversion?,” 2015. [Online]. Available: http://svnbook.red-bean.com/nightly/es/svn-ch-1-sect-1.html.

[22] K. D. la Cuadra, “¿Qué es GitHub?,” *Platzi*, 2017. [Online]. Available: https://platzi.com/blog/que-es-github/?gclid=EAIaIQobChMIoeu68ZjT4AIVB4TICh0wxgEAEAAYASAAEgJDyfD_BwE.

[23] F. Martinig, “StarUML - Open Source UML Tool,” 2011. [Online]. Available:

http://www.methodsandtools.com/tools/staruml.php.
[24] V. Paradigm, “Visual Paradigm Frequently Asked Question,” 2019. [Online]. Available: https://www.visual-paradigm.com/support/faq.jsp.
[25] Bizagi, “Bizagi Modeler.” [Online]. Available: https://www.bizagi.com/en/products/bpm-suite/modeler.
[26] D. Gewirtz, “G Suite: Everything you need to know before signing up for Google’s office suite,” *Cnet*, 2018. [Online]. Available: https://www.cnet.com/g00/how-to/g-suite-everything-you-need-to-know-before-signing-up-for-googles-office-suite/?i10c.ua=1&i10c.encReferrer=&i10c.dv=19.
[27] R. Devine, “What is Microsoft Office 365?,” 2015. [Online]. Available: https://www.windowscentral.com/what-microsoft-office-365.
[28] Atlassian, “Trello,” 2017. [Online]. Available: https://trello.com/en.
[29] Trello, “What is Trello?,” 2017. [Online]. Available: https://help.trello.com/article/708-what-is-trello.
[30] TeamGantt, “Why did we build TeamGantt?,” 2019. [Online]. Available: https://www.teamgantt.com/about-us.
[31] L. Curry, “What is Office 365 Planner? A Beginner’s Guide,” *Chorus*, 2019. [Online]. Available: https://www.chorus.co/resources/news/what-is-office-365-planner-a-beginners-guide.
[32] R. S. Pressman, *Ingeniería del software : un enfoque práctico*. McGraw-Hill, 2010.
[33] I. Sommerville, *Software Engineering*. Pearson, 2015.
[34] M. Sliger, “Agile estimation techniques,” 2012. [Online]. Available: https://www.pmi.org/learning/library/agile-project-estimation-techniques-6110.
[35] M. G. Software, “Planning Poker.” .
[36] D. Sinha, “Top 5 Scrum Estimation Techniques- Find Your Best Fit,” 2018. [Online]. Available: https://www.knowledgehut.com/blog/agile/top-5-scrum-estimation-techniques-find-your-best-fit.
[37] PsicoActiva, “Test de dominancia cerebral.” .
[38] M. G. Software, “Scrum Roles.” [Online]. Available: https://www.mountaingoatsoftware.com/agile/scrum/roles.
[39] University of Maryland, “Applied Scrum for Project Management,” *edX*, 2018. [Online]. Available: https://www.edx.org/course/applied-scrum-for-project-management?source=aw&awc=6798_1550554292_f321a3552b65c146c5d7991e351f291c&utm_source=aw&utm_medium=affiliate_partner&utm_content=text-link&utm_term=301045_https%3A%2F%2Fwww.class-central.com%2F.
[40] U. of Alberta, “Software Processes and Agile Practices,” *Coursera*, 2015. [Online]. Available: https://www.coursera.org/learn/software-processes-and-agile-practices.
[41] D. Merchante, “Powerexplosive,” 2013. [Online]. Available: https://powerexplosive.com.
[42] Bodybuilding.com, “Bodybuilding.com.” [Online]. Available: https://www.bodybuilding.com/.
[43] M. & Stregth, “Muscle & Stregth,” 2005. [Online]. Available: https://www.muscleandstrength.com.
[44] Drifty Co, “Ionic Framework,” 2018. [Online]. Available: https://ionicframework.com/docs.
[45] Google, “Firebase Guides,” 2014. [Online]. Available: https://firebase.google.com/docs/.
[46] Tutorialspoint, “Firebase Tutorial,” 2006. [Online]. Available: https://www.tutorialspoint.com/firebase/.
[47] S. R. ELearning, “Mobile UI and UX Design Tutorial,” 2015. [Online]. Available: https://www.youtube.com/watch?v=TYqUdDtTwis&list=PL5eJgcQ87sgfTfa339_OSu

AcdSyPSDSjJ&index=2.
[48] S. S. Caroline Buckey, “How to Use Git and GitHub,” *Udacity*, 2015. [Online]. Available: https://www.udacity.com/course/how-to-use-git-and-github--ud775.
[49] Google, “Google Docs: Free Online Documents for Personal Use.” [Online]. Available: https://www.google.com/docs/about/.
[50] Elsevier, “Mendeley - Reference Management Software & Researcher Network,” 2008. .
[51] S. LLC, “SmartDraw,” 2018. [Online]. Available: https://www.smartdraw.com/.
[52] TeamGantt, “Intuitive and Beautiful Project Planning.” [Online]. Available: https://www.teamgantt.com/.
[53] M. G. Software, “PlanningPoker.com,” 2014. [Online]. Available: https://www.planningpoker.com/about/.
[54] MKLab, “StarUML,” 2005. [Online]. Available: https://staruml.io/.
[55] C. A. Melanie Perkins, Cliff Obrecht, “Canva,” 2012. [Online]. Available: https://www.canva.com/.
[56] E. Oliveros, “WBS, Definiciones y Aplicaciones,” 2011, p. 28.
[57] O. B. School, “¿Qué es un diagrama de Gantt y para qué sirve?,” 2016. .
[58] ACIS, “Estudio de Salarios y Profesionales del Sector de Software y TI en Colombia 2017,” 2017.
[59] Opcionempleo, “Empleo Diseñador UX,” 2019. [Online]. Available: https://www.opcionempleo.com.co/empleo-disenador-ux.html.
[60] Elempleo, “Diseñador UI,” 2019. [Online]. Available: https://www.elempleo.com/co/ofertas-empleo/?&trabajo=ui.
[61] Computrabajo, “Diseñador UX/UI,” 2019. [Online]. Available: https://www.computrabajo.com.co/ofertas-de-trabajo/?sal=5&q=ui.
[62] Publimetro, “Aumentarán de 8 a 10 horas la jornada laboral,” 2017.
[63] W. S. Humphrey, *Introduction to the Team Software Process*, 5th ed. Massachusetss: Addison Wesley Logman, Inc., 2004.
[64] B. C, “Risks analysis in projects of investment Study case,” *Sci. Techhnica*, no. 38, pp. 309–314, 2008.
[65] Software Testing Help, “Types of Risks in Software Projects,” 2019. [Online]. Available: https://www.softwaretestinghelp.com/types-of-risks-in-software-projects/.
[66] Global Standard, *Fundamentos Para La Dirección De Proyectos (Guía Del Pmbok)*. 2014.
[67] Software Engineering Standards Committee of the and IEEE Computer Society, “IEEE Standard for Software Maintenance,” vol. 1998, 1998.