PLAN DE GESTIÓN DE PROYECTO DE SOFTWARE

PROYECTO

Instituto Tecnológico de Monterrey

Oracle Cloud

INTEGRANTES

Bella Elisabet Perales Meléndez y Alcocer - A00833423

Mariel Gisela Pérez Ferrusquía - A00832811

Daniela Nuño Martinez - A01177702

Andrés Fernando Garza Garcia - A01138704

Marcelo García Pablos Vélez - A00815371

1. Visión del Proyecto
    1.1. Descripción General
    1.2. Propósito, alcance, objetivos
        1.2.1 Propósito
        1.2.2. Alcance
        1.2.3. Objetivo
    1.3. Asunciones y Restricciones
    1.4. Entregables
2. Backlog
    2.1. Objetivo
    2.2. Historias
3. Plan de comunicación
    3.1. Métodos y Herramientas
        3.1.1. Herramientas de Gestión de Proyectos
        3.1.2. Medios de comunicación
    3.2. Roles y Responsabilidades
    3.4. Calendario
4. Estrategia de desarrollo
    4.1. Metodología de desarrollo
    4.2. Proceso de desarrollo
    4.3. Roles y responsabilidades
5. Plan de recursos (gestión del proceso técnico)
    5.1. Presupuesto y Finanzas
        5.1.1 Distribución del Presupuesto:
    5.2. Recursos humanos
        5.2.1 Roles
        5.2.2 Modalidades de Trabajo
    5.3. Infraestructura y Herramientas
    5.4. Desarrollo y operaciones
    5.5. Etapas del proyecto
    5.6. Seguridad
    5.7. Documentación y colaboración
    5.8. Reserva para Imprevistos
6. Plan de riesgos
    6.1. Matriz de identificación de riesgos
    6.2. Matriz de probabilidad - impacto
    6.3. Planificación de respuestas a riesgos
7. Plan de valor ganado
8. Métricas ágiles

1. Visión del Proyecto

1.1. Descripción General

El proyecto de “Oracle Java Bot” busca automatizar las tareas del desarrollador y brindar visibilidad al manager del equipo. Su principal objetivo es mejorar la productividad y la visibilidad de las actividades disponibles dentro del equipo de desarrollo de software de Oracle, siguiendo los principios de DevOps. Un desarrollador podrá visualizar, crear y eliminar tareas, mientras que un manager podrá visualizar las actividades disponibles de cada desarrollador. Además, se espera proporcionar un medio de comunicación eficiente entre los miembros del equipo y su manager utilizando la plataforma de comunicación Telegram.

1.2. Propósito, alcance, objetivos

1.2.1 Propósito

- Automatizar las tareas del equipo de desarrollo.
- Proporcionar visibilidad de las actividades de cada miembro del equipo al manager.
- Mejorar la productividad del equipo en un 20%.
- Permitir a los desarrolladores agregar/eliminar/editar tareas personales.

1.2.2 Alcance

Desarrollar e implementar un ChatBot que permita a los desarrolladores revisar, agregar, eliminar y marcar tareas, así como proporcionar al manager una visión general de las tareas del equipo. Se busca implementar el proyecto en los equipos de trabajo de Oracle.

1.2.3 Objetivo

Implementar una solución que mejore la productividad y la visibilidad del equipo de desarrollo de software en un 20%.

1.3. Asunciones y Restricciones

- Se utilizará Telegram como único medio de comunicación entre miembros del equipo.
- Se utilizarán herramientas y servicios específicos de Oracle incluyendo Oracle Cloud Infrastructure.
- Debe cumplir con los estándares de seguridad utilizados por los servicios de Oracle.
- Se manejan dos modalidades de trabajo: híbridas y remotas.

1.4. Entregables

- Un chatbot funcional en Java.
- Documentación sobre el desarrollo y acuerdos del proyecto.
- Manual del usuario.
- Repositorio del código público en GitHub.

2. Backlog

2.1. Objetivo

El objetivo del proyecto “Oracle Java Bot” es mejorar la productividad y visibilidad de las tareas del equipo de desarrollo de software de Oracle, incrementando la eficiencia en un 20%.

2.2. Epic: Automatizar las tareas del equipo.

2.2.1 Feature #1: Gestión de tareas del desarrollador.

2.2.1.1 Historia de Usuario #1: Como desarrollador, quiero poder visualizar mis tareas para mantenerme al tanto de mis responsabilidades.

2. 2. 1.2 Historia de Usuario #2: Como desarrollador, quiero poder crear tareas para mantener un orden en mi trabajo.

2.2.1.3 Historia de Usuario #3: Como desarrollador, quiero poder eliminar tareas para evitar modificaciones.

2.2.1.4 Historia de Usuario #4: Como desarrollador, quiero poder marcar una tarea como completa.

2.2.1.5 Historia de Usuario #5: Como desarrollador, quiero poder marcar una tarea como no-completa para poder volver a trabajar en ella.

2.2.2 Feature #2: Supervisión de tareas por el manager.

2. 2. 2.1 Historia de Usuario #1: Como manager, quiero visualizar las tareas que cada uno de mis desarrolladores realizan para llevar un seguimiento del progreso que el equipo tiene.

2.2.2.2 Historia de Usuario #2: Como manager, quiero poder crear modificar las tareas de un desarrollador en específico.

3. Plan de Comunicación

3.1. Métodos y herramientas

3.1.1. Herramientas de Gestión de Proyectos

- GitHub proyectos: Para asignar tareas, hacer seguimiento del progreso y gestionar el backlog del proyecto. Todos los miembros del proyecto tendrán acceso para ver tareas y actualizaciones.
- Documentos Compartidos (Google Drive): Para almacenar y compartir documentación del proyecto, incluyendo el plan de proyecto, SRS, y documentación técnica, asegurando que todos tengan acceso a la información más reciente.

3.1.2. Medios de comunicación

- Correo Electrónico: Para comunicaciones oficiales, distribución de documentos importantes y actualizaciones generales del proyecto.
- Videoconferencias (Zoom): Para reuniones semanales de forma remota, revisiones de sprint y sesiones de planificación. También se utilizará para discusiones importantes que requieran una deliberación en profundidad.

3.2. Roles y responsabilidades

- Scrum master: Facilitar las ceremonias de Scrum (Daily Stand-ups, Sprint Planning, Sprint Review, Sprint Retrospective), ayudar al equipo a remover impedimentos, y asegurar que se sigan las prácticas de Scrum.
- Product owner: Definir y priorizar el Backlog del Producto, asegurando que las tareas estén claramente definidas y alineadas con los objetivos del proyecto. Comunicar las

necesidades y cambios de prioridades al equipo. Proporcionar actualizaciones regulares al equipo sobre la visión del proyecto y recoger feedback sobre el desarrollo en curso.

- Equipo de desarrollo: Implementar las funcionalidades definidas en el Backlog del Sprint con las herramientas computacionales, mantener una comunicación abierta sobre el progreso y los desafíos, y contribuir al continuo mejoramiento del equipo.

3.3. Calendario

- Daily Stand-up (Diario): 15 minutos a las 3pm presencial en el tec para que el equipo sincronice actividades y discuta obstáculos. Si no se puede hacer reunión presencial, se optará por usar la herramienta de zoom en el mismo horario.
- Sprint Planning (Inicio de cada Sprint): Sesión de planificación para definir el trabajo a realizar en el próximo Sprint cada 3 semanas que es la duración en la que deben de estar completadas todas las historias de usuario del sprint. Este se realizará por zoom o presencial dependiendo la disponibilidad de los integrantes a participar.
- Sprint Review (Final de cada Sprint): un día antes de cada sprint planning para demostrar el trabajo realizado y recoger feedback del Product Owner y stakeholders. Este se realizará por zoom o presencial dependiendo la disponibilidad de los integrantes a participar.
- Sprint Retrospective (Final de cada Sprint): el mismo día del sprint review (al finalizarlo) para reflexionar sobre el Sprint pasado e identificar oportunidades de mejora. Este se realizará por zoom o presencial dependiendo la disponibilidad de los integrantes a participar.
- Desarrollo sprint: Los desarrolladores podrán realizar su trabajo durante el día a la hora que se les sea más fácil, siempre y cuando terminen sus avances para el daily stand-up.

4. Estrategia de desarrollo

4.1. Metodología de desarrollo

- Uso de metodologías Ágiles SCRUM:
  Se utilizará una metodología ágil para el desarrollo del proyecto, con iteraciones cortas y entregas incrementales. Se realizarán reuniones regulares de revisión y planificación para garantizar la alineación con los objetivos del proyecto y responder a los cambios en los requisitos.

4.2. Proceso de desarrollo

4.2.1 Definición de Requisitos

- La recopilación de los requisitos se llevará a cabo en múltiples sesiones de entrevista con el socio formador. Inicialmente hay una sesión introductoria al proyecto de forma general y posteriormente habrá múltiples entrevistas en las que los requerimientos tendrán modificaciones discretas de acuerdo a las necesidades del socio. Se definen los requerimientos funcionales y no funcionales para su documentación en el SRS que será la base para el desarrollo del proyecto.

- La metodología SCRUM permitirá que los requerimientos iniciales puedan ser ajustados conforme avanza el proyecto de ser necesario con una adaptación más fácil y rápida.

4.2.2 Diseño del Sistema

- . Se diseña la arquitectura base del sistema con todos los componentes necesarios para su ejecución:

    i. UI/UX: Debe ser un diseño fácil de usar para los usuarios basado en la herramienta principal Telegram. Se define la estructura del chat y como sus funcionalidades principales serán realizadas a partir de lo que haga el usuario (listas de opciones, reconocimiento de texto, etc).
    ii. Modelo de datos: Se deben diseñar los esquemas de bases de datos que se utilizarán para soportar las funcionalidades del bot, desde datos relacionados a los usuarios y gerentes hasta datos de las tareas y proyectos incluyendo su recopilación, almacenamiento y manejo. Los servicios de Oracle Cloud Infrastructure son los primordiales en cuanto a manejo de datos en la nube, este paso se enfoca principalmente en el diseño de esta misma base de datos.
    iii. Seguridad: Elección de medidas de seguridad que se integrarán al sistema para incluir una autenticación eficaz y confiable, así como encriptado de chats por usuario y gerente.
    iv. Definición de APIs: Se definen las APIs a emplear dentro de los servicios internos y, de ser necesario, servicios externos.

4. 2. 3 Implementación

- Desarrollo del sistema: Con base en el diseño del sistema se procede a desarrollar el código y otros componentes necesarios para el funcionamiento correcto del ChatBot. Durante el desarrollo se revisarán nuevamente los requerimientos y el equipo se asegurará de que cada requerimiento tanto funcional como no funcional se cumpla tomando en cuenta casos de uso y las historias de usuario.

4.2.4 Prueba

- Se establecerá un tiempo específico para realizar pruebas funcionales unitarias, pruebas de integración, de seguridad, rendimiento, compatibilidad, usabilidad y documentación. Todas ellas explicadas de forma explícita en el documento “Plan de calidad”.

4.2.5 Entrega

- Despliegue Continuo: Utilizando CI/CD, el equipo automatizará el proceso de despliegue para entregar incrementos del software de manera eficiente al ambiente de producción, permitiendo una entrega continua de valor al cliente final.

- Revisión y Feedback del Socio Formador: Cada entrega será revisada junto con el socio formador para recoger comentarios y ajustar el producto según sea necesario.

4.2.6 Mantenimiento

- Soporte Post lanzamiento: Tras la implementación inicial, el equipo brindará soporte para el ChatBot, abordando cualquier problema técnico o error que surja durante su uso real.
- Mejoras Continuas: Basándose en el feedback de los usuarios y las tendencias tecnológicas emergentes, el equipo se encargará de recolectar la información para germinar mejoras y planificar e implementar la solución en el ChatBot, garantizando su relevancia y efectividad a largo plazo.

5. Plan de Recursos

5.1 Presupuesto y Finanzas

El presupuesto total estimado del proyecto es de $625,000 MXN.

5.1.1 Distribución del Presupuesto:

- Infraestructura y Herramientas de Desarrollo: 40%
- Desarrollo y Pruebas: 30%
- Operaciones y Monitoreo: 20%
- Reserva para imprevistos: 10%

5.2 Recursos Humanos

5.2.1 Roles

- Desarrolladores Java (3)
- Especialista en DevOps y CI/CD (1)
- Manager del Proyecto (1)

5.2.2 Modalidades de Trabajo

- Remota y Híbrida

5.3 Infraestructura y Herramientas

- Cloud: Oracle Cloud Infrastructure
- Base de Datos: Oracle Autonomous Database
- Contenedores: Kubernetes, Docker
- Desarrollo:
  - Lenguaje: Java
  - Framework: Spring Boot
  - Microservicios, API Gateway, Container Registry

5.4 Desarrollo y Operaciones (DevOps)

- Integración y Despliegue Continuo: Automatización de ciclos de desarrollo, integración continua (CI), despliegue continuo (CD)
- Ambientes de Desarrollo: Aprovisionamiento con Infrastructure as Code

5.5 Etapas del Proyecto

Phase 1: MVP y Infraestructura para el Desarrollo
Phase 2: Desarrollo del Código
Phase 3: Construcción
Phase 4: Pruebas
Phase 5: Lanzamiento
Phase 6: Despliegue
Phase 7: Operación
Phase 8: Monitoreo

5.6 Seguridad

- Componentes: Asegurar todos los componentes de la arquitectura de software

5.7 Documentación y Colaboración

- Repositorio: Github.com para el código y documentación
- Planificación Estratégica: Uso de Strategic Planner y Jira para seguimiento de tareas

5.8 Reserva para Imprevistos

- Fondo de Reserva: Para cubrir gastos no previstos o ajustes en el presupuesto

6. Plan de Riesgos

6.1. Matriz de Identificación de Riesgos

ID	Evento	Probabilidad	Consecuencia	Riesgo	Calificación
R1	Problemas con la integración de distintos elementos del proyecto	Media	Mayor	12	12
R2	Un integrante del equipo no cumple con su avance	Alta	Moderada	12	12
R3	El socio formador cambia requerimientos del sistema	Media	Mayor	12	12
R4	No disponibilidad del software o hardware necesario para el desarrollo	Baja	Máxima	10	10
R5	Se realizan pruebas incompletas por situaciones adversas	Baja	Moderada	6	6
R6	Requerimientos funcionales o no funcionales iniciales incompletos	Alta	Mayor	16	16
R7	Comunicación faltante con el socio formador	Baja	Mayor	8	8
R8	Se agotan los recursos para continuar con el proyecto	Alta	Mayor	16	16
R9	Subestimación del tiempo necesario para cada avance	Media	Moderada	9	9
R10	Falta de comunicación efectiva en el equipo	Muy baja	Mayor	4	4
R11	Se agregan requerimientos de forma contínua durante el desarrollo	Alta	Mayor	16	16

6.2. Matriz de Probabilidad - Impacto

		Consecuencia				
		Minima	Menor	Moderada	Mayor	Maxima
Probabilidad		1	2	3	4	5
Muy alta	5					
Alta	4			R2	R6, R8, R11	
Media	3			R9	R1, R3	
Baja	2			R5	R7	R4
Muy baja	1				R10	

Nivel de riesgo	Color
Riesgo aceptable	
Riesgo tolerable	
Riesgo alto	
Riesgo extremo	

6.3. Planificación de respuestas a riesgos

Riesgo	Análisis
R1	- Realizar pruebas de integración periódicas durante todo el ciclo de vida del proyecto. - Implementar estándares de codificación y diseño para garantizar la cohesión y la interoperabilidad entre los distintos componentes del sistema.
R2	- Establecer objetivos y plazos claros para cada miembro del equipo. - Realizar reuniones de seguimiento periódicas para identificar cualquier problema temprano y ofrecer ayuda si es necesario. - Considerar la reasignación de tareas o la capacitación adicional si un miembro del equipo está luchando con su trabajo asignado.
R3	- Establecer un proceso de gestión de cambios formal que requiera una evaluación del impacto de los cambios propuestos antes de su implementación. - Mantener una comunicación abierta y regular con el socio formador para comprender las necesidades cambiantes y anticipar posibles cambios en los requisitos.
R4	- Realizar una planificación anticipada de los recursos necesarios y adquirirlos con suficiente antelación. - Identificar alternativas o soluciones de contingencia en caso de que los recursos no estén disponibles según lo planeado.
R5	- Desarrollar planes de contingencia para situaciones adversas que puedan afectar las pruebas. - Automatizar las pruebas siempre que sea posible para aumentar la eficiencia y garantizar la cobertura adecuada.

R6	- Realizar un análisis detallado de los requisitos con todas las partes interesadas para garantizar que se capturen de manera completa y precisa. - Utilizar metodologías ágiles que permitan la adaptación a medida que se descubren nuevos requisitos o se realizan cambios.
R7	- Establecer canales de comunicación claros y regulares con el socio formador. - Designar un punto de contacto específico para facilitar la comunicación y garantizar que las preocupaciones se aborden de manera oportuna.
R8	- Realizar una gestión proactiva de los recursos para identificar y mitigar posibles problemas de asignación anticipadamente. - Considerar la reasignación de recursos o la búsqueda de recursos adicionales si es necesario para garantizar la continuidad del proyecto.
R9	- Utilizar técnicas de estimación de tiempo más precisas, como la descomposición del trabajo en tareas más pequeñas y la consulta con expertos en la materia. - Realizar un seguimiento continuo del progreso para identificar y abordar desviaciones en el cronograma lo antes posible.
R10	- Fomentar un ambiente de trabajo abierto y transparente donde los miembros del equipo se sientan cómodos expresando sus ideas y preocupaciones. - Utilizar herramientas de colaboración y comunicación, como plataformas de gestión de proyectos y reuniones regulares, para facilitar la comunicación entre los miembros del equipo.
R11	- Implementar un proceso de gestión de cambios efectivo que evalúe el impacto de cada nuevo requerimiento en el alcance, el tiempo y el presupuesto del proyecto. - Educación y sensibilización sobre la importancia de establecer requisitos claros y estables desde el principio del proyecto.

7. Plan de Valor Agregado

7.1. Descripción y objetivo

La metodología EVM (Earned Value Management) describe un plan para la evaluación del progreso y rendimiento del proyecto. Se toman en cuenta varios factores, incluyendo el alcance del proyecto, el cronograma y los costos seccionados y totales del proyecto para determinar métricas importantes y cómo se medirá el progreso. En un plan de valor ganado se deben establecer primordialmente los costos estimados en todos los aspectos y posteriormente incluir variables para una medición objetiva de índices relevantes. A continuación se incluyen subsecciones enfocadas en cada aspecto del plan de valor ganado.

7.2. Parámetros

7.2.1 Presupuesto Total:

El presupuesto total es de $625,000.

7.2.2 SCRUM:

Ver SCRUM adjunto en la sección de estrategia de desarrollo.

7.2.3 Cronograma:

Ver calendario semanal adjunto en la sección 3.3. El tiempo total estimado para la finalización del proyecto es de 3.5 meses.

7.3. Cálculo de métricas

Las métricas son valores que se estiman tomando el presupuesto total como base. Es importante considerar que la mayor parte del presupuesto total para un proyecto como el actual es destinado a los sueldos del equipo, incluyendo los desarrolladores y el gerente del equipo (considerando que en nuestro ejemplo una persona actuará como gerente, pero de no ser así los sueldos serían equitativos entre los desarrolladores por la experiencia de cada quien, la cual es igual. En caso de tener niveles distintos de desarrolladores los sueldos podrían ser variables). Además de los sueldos del equipo es necesario estimar costos variables de otros aspectos del proyecto que podrán dividirse como se observa a continuación:

- Sueldos = 70%
    - $437,500
- Otros costos = 30% (dividido a continuación)
    - Infraestructura y herramientas de desarrollo = 40%
        - $75,000
    - Pruebas = 30%
        - $56,250
    - Operaciones y mantenimiento = 20%
        - $37,500
    - Reservas = 10%
        - $18,750

Las métricas de importancia incluyen el planned value, earned value y actual cost… el PV describe el valor mensual del proyecto, el EV se calcula dependiendo del porcentaje del proyecto que estará completo en un rango de tiempo y el AC es el estimado del costo real del proyecto mensual.

- PT = $625,000
- Sueldos y equipo para los desarrolladores: 625,000 * 0.7 = $437,500
    - Desarrolladores: $24,000
    - Gerente: $29,000
- Métricas hipotéticas por mes:
    - PV (Planned value): 625,000/3.5 meses ≈ 178,000
    - EV (Earned value): 0.28 * 625,000 ≈ 175,000
        - Suponiendo que el 28% del proyecto va a estar completo para el primer mes
    - AC (Actual cost): $180,000

- Estimado del costo real total en el primer mes

7.4. Variaciones a considerar

- Las variaciones son medidas que ayudan a determinar que tanto probablemente el equipo pueda ir sobre presupuesto o detrás del presupuesto. Son valores para tener una idea de lo que podría pasar en caso de que el porcentaje completo del proyecto o el cronograma salgan fuera de lo esperado en un rango de tiempo. Si las varianzas son negativas el proyecto estaría inclinándose a un sobre presupuesto, en cambio si las varianzas son positivas el presupuesto es más que suficiente de acuerdo al avance del proyecto.
- CV (Cost Variance) = EV - AC = 175,000 - 180,000 = -5,000
    - Esto denota que el proyecto estaría sobre el presupuesto, por lo cual el AC debe disminuir o el EV debe aumentar (completando más porcentaje del proyecto por mes).
- SV (Schedule Variance) = EV - PV = 175,000 - 178,000 = -3,000
    - Esto significaría que el proyecto va atrasado en el cronograma por el valor negativo del cálculo. En este caso se buscaría aumentar el EV con un mayor avance mensual.

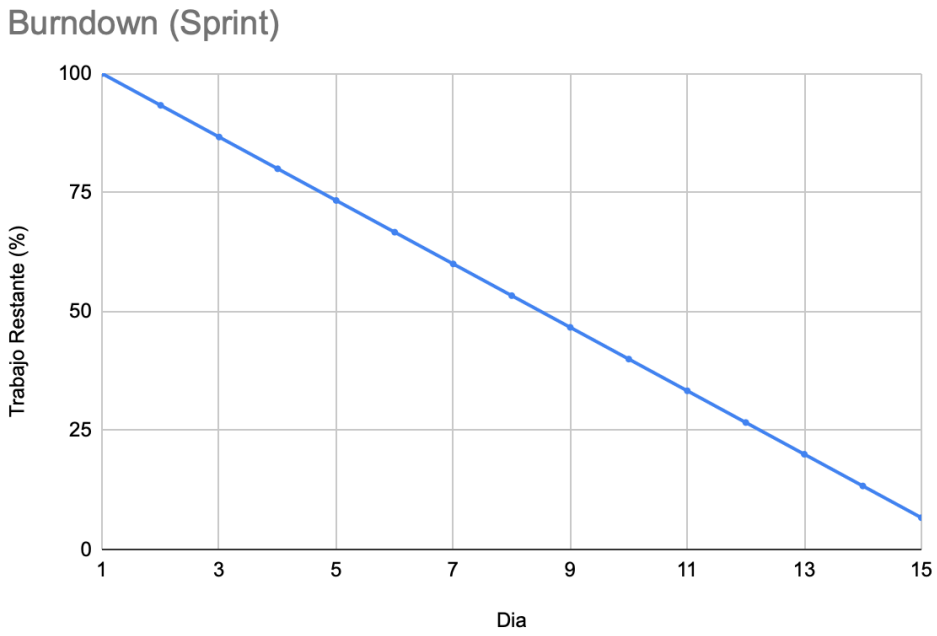
7.5. Índices

- Los índices toman en cuenta las métricas establecidas anteriormente para medir la eficiencia del desarrollo del proyecto ya que analizan el rendimiento del costo del proyecto y del cronograma real. A continuación se muestran los cálculos realizados para obtener los dos índices primordiales y una breve descripción de sus significados.
- CPI (Cost Performance Index) = EV/AC = 175,000 / 180,000 = 0.97
    - El CPI menor a 1 indica que el proyecto es menos eficiente en costos de lo planificado.
- SPI (Schedule Performance Index) = EV/PV = 175,000/178,000 = 0.98
    - El SPI menor a 1 indica que el proyecto está avanzando más lento de lo deseado.

8. Métricas ágiles

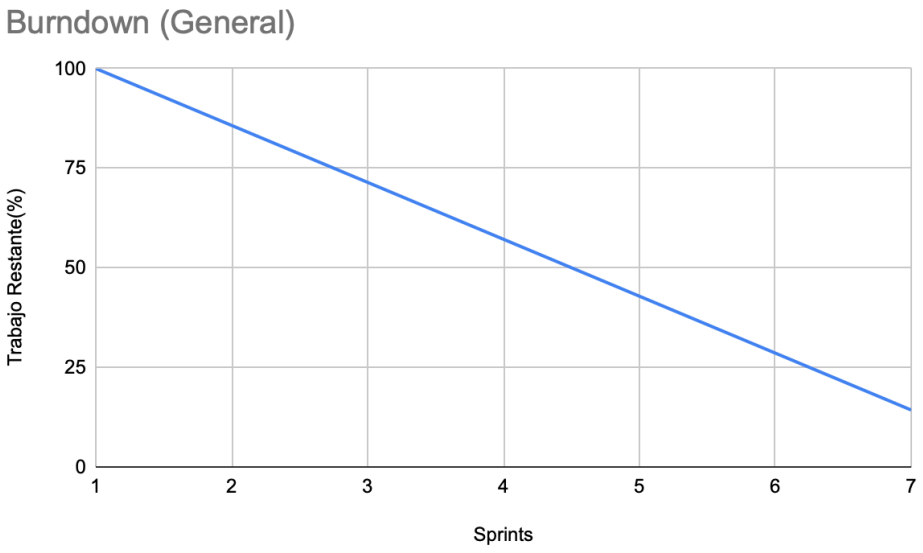
8.1 Burndown Chart

- El burndown chart es una métrica que se puede utilizar en metodologías ágiles, específicamente en SCRUM para denotar el tiempo restante vs. el trabajo realizado. En el eje horizontal se observa el tiempo y en el eje vertical el trabajo restante. Se traza una línea diagonal inicial que empieza en la esquina superior izquierda del gráfico y termina en la esquina inferior derecha, la cual define como debería avanzar el trabajo a través del tiempo. Conforme avanza el proyecto se agregan puntos del resultado obtenido y se compara con la línea inicial, demostrando si el progreso es ideal o faltante. En este caso se hacen gráficas para cada sprint y una gráfica general con las líneas ideales integradas, conforme avance el proyecto se actualizarán las gráficas.



En el caso de cada sprint, la duración es de 15 días y en el eje vertical se muestra el porcentaje de trabajo restante. La línea azul denota el progreso en caso de que todos los días se realice la misma cantidad de trabajo. El tiempo total es de 3.5 meses (105 días), por lo que cada 3 semanas se aplica la gráfica de Sprint.

- Para una metodología ágil en ocasiones es conveniente tener una gráfica general para todo el proyecto seccionada por sprints:



En este caso se sabe que en los 3.5 meses habrá un total de 7 sprints, por lo que el eje horizontal denota sprints y el eje vertical el porcentaje de trabajo a realizar en cada sprint. La línea azul denota el progreso ideal si en cada sprint se completa la misma cantidad de trabajo.

8.2 Velocity chart

- Una métrica esencial en la metodología ágil es una gráfica de velocidad para tener una medición del trabajo realizado y determinar la estructura del trabajo posterior. El eje horizontal representa los sprints consecutivos (en este caso serán 7) y el eje vertical representa el porcentaje de trabajo realizado. Conforme avanza el desarrollo se incluyen puntos (o barras) para trazar una línea y determinar la velocidad del progreso del proyecto. Con esta gráfica se pueden hacer modificaciones para aumentar o disminuir el trabajo a realizar el próximo sprint de ser necesario. A diferencia del Burndown, el Velocity ideal tendrá una línea diagonal que inicia en la esquina inferior izquierda y termina en la esquina superior derecha. Sin haber iniciado el proyecto no es posible mostrar una gráfica de velocidad.