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. Introducción
2. Visión del Proyecto
    2.1. Descripción General
    2.2. Contexto, Antecedentes
        2.2.1 Contexto
        3.2.2. Antecedentes
    2.3. Propósito, alcance, objetivos
        2.3.1 Propósito
        2.3.2. Alcance
        2.3.3. Objetivo
    2.4. Asunciones y Restricciones
    2.5. Entregables
    2.6. Análisis Costo Beneficio
3. Organización del Proyecto
    3.1. Roles del Equipo y Funciones
    3.2. Stakeholders y Funciones
4. Backlog
    4.1. Objetivo
    4.2. Epic, Feature, Historia de Usuario
    4.3. Epic, Feature, Historia de Usuario
5. Plan de comunicación
    5.1. Propósito y Enfoque
    5.2. Métodos y Herramientas
        5.2.1. Herramientas de Gestión de Proyectos
        5.2.2. Medios de comunicación
    5.3. Roles y Responsabilidades
    5.4. Matriz de comunicación
    5.5. Calendario
6. Estrategia de desarrollo
    6.1. Metodología de desarrollo
        6.1.1. Ágil SCRUM
        6.1.2. Prácticas SCRUM
        6.1.3. Métricas y Reportes
    6.2. Proceso de desarrollo
        6.2.1. Definición de Requisitos
        6.2.2. Diseño del Sistema
        6.2.3. Implementación
        6.2.4. Prueba
        6.2.5. Entrega
        6.2.6. Mantenimiento
7. Manejo de Cronograma
    7.1. Calendario de Ceremonias SCRUM

1. Introducción

El Bot de Java de Oracle tiene como objetivo automatizar tareas de desarrolladores y proporcionar visibilidad al gerente del equipo. El proyecto del Bot de Java de Oracle

representa una iniciativa significativa dentro del ecosistema de desarrollo de software de Oracle, con el objetivo de adoptar los principios de la filosofía DevOps. Proporcionando orientación y estableciendo estándares que los servicios de Oracle deben cumplir, este documento sirve como un plan integral, detallando las funcionalidades, requisitos y parámetros del Bot de Java de Oracle, así como los planes de desarrollo con los que contará el proyecto.

2. Visión del Proyecto

2.1. 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.

2.2. Contexto, antecedentes

2.2.1 Contexto

En la actualidad, un porcentaje mínimo de proyectos de desarrollo de software son éxito puro, con un 17%. El resto de estos proyectos de software se entregan incompletos, tarde, fuera de presupuesto, sin la satisfacciones de los clientes o son pérdidas totales. Existen varias razones por las cuales los proyectos de software son considerados como fracasos debido a los errores. Entre estas, se incluyen las estimaciones inexactas o excesivamente optimistas, un mal control de calidad, un control de cambios inadecuado para el proyecto, un mal monitoreo del tiempo y progreso engañoso, y pasar por alto problemas técnicos sencillos.

2.2.2 Antecedentes

Oracle, actualmente, tiene como objetivo mejorar la productividad y visibilidad de las actividades de cada miembro del equipo de desarrollo. Los equipos de trabajo funcionan con dos modalidades: híbridas y remotas, por lo que el servicio que buscan debe ser capaz de soportar estas modalidades. En la actualidad, los nuevos servicios de software para los clientes de Oracle utilizan la filosofía DevOps, por lo que el servicio solicitado debe apegarse a dicha filosofía.

2.3. Propósito, alcance, objetivos

2.3.1 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.

2.3.2 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.

2.3.3 Objetivo

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

2.4. 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.

2.5. Entregables

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

2.6. Análisis Costo Beneficio

Para la elaboración de este proyecto se estima un uso de $625,000 MXN. Considerando esta como una inversión inicial y general para el desarrollo completo del sistema, incluidos los su eldos de cada integrante del equipo, todas las etapas de inicio a fin del desarrollo del proyecto y los servicios necesarios para su correcta ejecución. Ver el Plan de Recursos para conocer más a detalle las funciones, requisitos y ganancias del proyecto.

3. Organización del Proyecto

3.1. Roles del Equipo y Funciones

A continuación se presentan los miembros del equipo y sus funciones específicas dentro del desarrollo del sistema:

a. Bella Elisabet Perales Meléndez y Alcocer (Manager del Proyecto):
   i. Organizar las principales actividades semanales.
   ii. Entablar una comunicación efectiva con los stakeholders.
   iii. Realizar y dirigir sesiones semanales con el equipo de trabajo.

b. Mariel Gisela Pérez Ferrusquía (Desarrolladora del Bot):
   i. Crear las funciones del Chat Bot en Telegram.
   ii. Realizar las pruebas necesarias para verificar el correcto funcionamiento del mismo.
   iii. Desarrollar un flujo de mensajes atractivos e intuitivos para los usuarios.

c. Daniela Nuño Martinez (Desarrolladora de Web):
    i. Crear las funciones principales de la página web.
    ii. Desarrollar vistas atractivas e intuitivas para los usuarios.
    iii. Generar pruebas para comprobar la ejecución del sistema.

d. Andrés Fernando Garza Garcia (Especialista en DevOps y CI/CD):
    i. Realizar pruebas exhaustivas al Chat Bot y página web en la infraestructura de la nube.
    ii. Integrar los sistemas en la infraestructura de la nube.
    iii. Migrar el sistema a seguir la filosofía DevOps.

e. Marcelo García Pablos Vélez (Especialista en DevOps y CI/CD)
    i. Migrar el sistema a la infraestructura en la nube.
    ii. Enlazar el GitHub del equipo con la infraestructura de la nube.
    iii. Realizar la automatización de las pruebas.

3.2. Stakeholders y Funciones

a. Desarrolladores de Software de Oracle.
    i. Utilizar el Chat Bot para gestionar sus tareas personales.
    ii. Proporcionar retroalimentación sobre la usabilidad y funcionalidad.

b. Managers de Equipos de Desarrollo de Software de Oracle:
    i. Utilizar el Chat Bot para monitorear las tareas de su equipo.
    ii. Proporcionar retroalimentación sobre la usabilidad y funcionalidad.
    iii. Promover el uso del Chat Bot dentro de su equipo.

c. Comité Directivo del Proyecto “Oracle Chat Bot”:
    i. Brindar retroalimentación sobre el Chat Bot y la interfaz web.
    ii. Evaluar el progreso del proyecto y tomar decisiones sobre la organización.
    iii. Mantener comunicación continua y efectiva con el equipo de desarrollo.
    iv. Realizar sesiones semanales para observar el progreso.

d. Clientes de Oracle
    i. Proporcionar retroalimentación sobre la mejora en la productividad de los equipos de desarrollo.
    ii. Recibir beneficios sobre el uso del Chat Bot con entregas más eficientes.

4. Backlog

4.1. 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%.

4.2. Epic #1: Incrementar la productividad de las tareas del equipo.

4.2.1 Feature #1: Gestión de tareas del desarrollador.

4.2.1.1 Historia de Usuario #1: Como desarrollador, quiero poder visualizar mis tareas para mantenerme al tanto de mis responsabilidades.
4.2. 1.2 Historia de Usuario #2: Como desarrollador, quiero poder crear tareas para mantener un orden en mi trabajo.
4.2.1.3 Historia de Usuario #3: Como desarrollador, quiero poder eliminar tareas para evitar modificaciones.
4.2.1.4 Historia de Usuario #4: Como desarrollador, quiero poder marcar una tarea como completa.
4.2.1.5 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.

4.3. Epic #2: Implementación de DevOps.

4.3.1 Feature #3: Integración con herramientas de CI/CD.

4.3.1.1 Historia de Usuario #1: Como desarrollador, quiero integrar nuestro código con herramientas de CI/CD para asegurar la calidad del software mediante pruebas automáticas y despliegues eficientes.

4.3.2 Feature #4: Automatización de despliegues.

4.3.2.1 Historia de Usuario #1: Como desarrollador, quiero que los despliegues a producción sean automatizados para reducir errores manuales y agilizar el proceso de release.

5. Plan de Comunicación

5.1. Propósito y Enfoque

- El principal propósito es definir las estrategias, métodos, herramientas y mecanismos que se estarán utilizando para garantizar una comunicación efectiva entre todos los involucrados en el proyecto.
- Se fomentará una comunicación abierta, honesta y proactiva entre todos los miembros del equipo y los stakeholders.

5.2. Métodos y herramientas

5.2.1. 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.

5.2.2. 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.

5.3. 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.

5.4. Matriz de comunicación

- Informe de Estado del Proyecto: Al Comité Directivo del Proyecto, se realizan sesiones semanales para informar del avance.
- Minutas de Reunión: A todos los miembros del equipo, se les dará una minuta después de cada reunión.
- Actualizaciones de Tareas: A todos los miembros, se les recordará de las actividades diarias en GitHub Projects.

5.5. 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.

6. Estrategia de desarrollo

6.1. Metodología de desarrollo

6.1.1 Ágil 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.

6.1.2 Prácticas SCRUM:

- Daily Stand-up: Reuniones diarias de 15 minutos donde cada miembro del equipo menciona su progreso, identificando áreas de oportunidad.
- Sprint Review: Al finalizar un sprint, se realizará una reunión para revisar el trabajo completado y capturar los comentarios de los stakeholders.
- Sprint Retrospective: Al finalizar un Sprint Review, se realizará una reunión para reflexionar sobre el Sprint, identificando áreas de oportunidad.

6.1.3 Métricas y Reportes:

Las siguientes métricas se utilizaran para monitorear el progreso del proyecto y reportan ante el Comité Directivo del Proyecto.

- Burndown Chart: Gráfica que muestra el trabajo restante en el Sprint comparado con el tiempo disponible. Para ver más observar el apartado Métricas Ágiles.
- Velocity: Medida de la cantidad de trabajo que el equipo puede completar en un Sprint. Para ver más observar el apartado Métricas Ágiles.

6.2. Proceso de desarrollo

6.2.1 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.

6.2.2 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.

6.2.3 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.

6.2.4 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”.

6.2.5 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 Oracle: Cada entrega será revisada junto con el socio formador para recoger comentarios y ajustar el producto según sea necesario.

6.2.6 Mantenimiento

- Soporte Post lanzamiento: Tras la implementación inicial, el equipo brindará soporte para el Chat Bot, 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 Chat Bot, garantizando su relevancia y efectividad a largo plazo.

7. Manejo de Cronograma

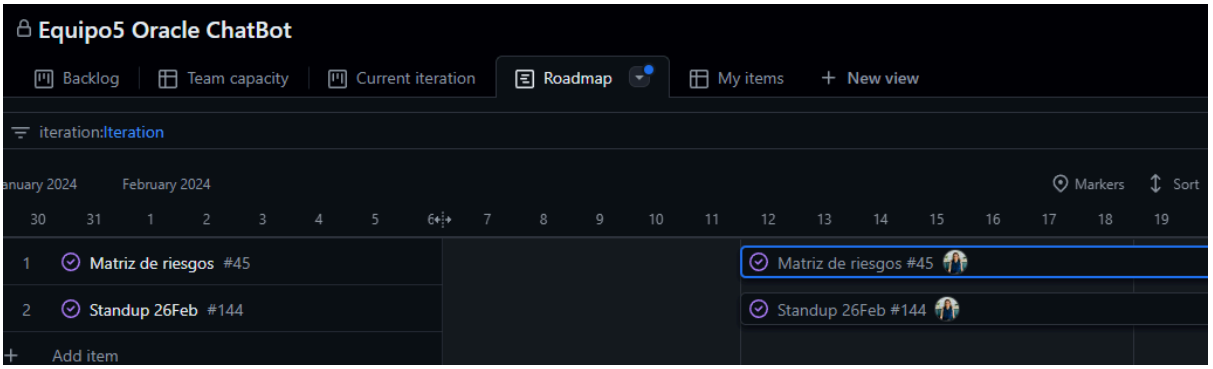
7.1 Calendario de Ceremonias SCRUM

- Sprint Planning: Cada 2 semanas, el primer día del nuevo Sprint.
- Daily Stand-Up: Diariamente antes de clase, (1:00 pm o 3:00 pm).
- Sprint Review: Un día antes del Sprint Planning siguiente.
- Sprint Retrospective: Mismo día que el Sprint Review, después de la revisión.

7.2 Calendario de Epic, estimación y cierre de proyecto

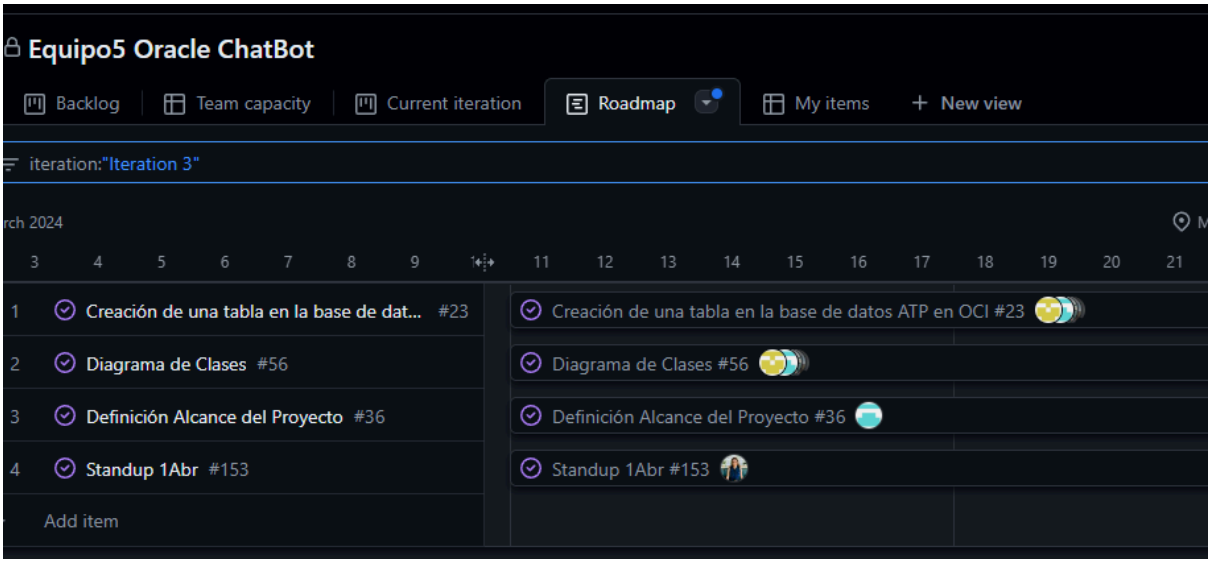
- El proyecto tiene un estimado de 15 semanas de duración, dividido en las fases de planeación, desarrollo y pruebas.
- Se busca finalizar el proyecto el 13 de junio del 2024.
- La duración de cada Sprint es de 2 semanas, teniendo alrededor de 7 sprints durante 15 semanas, por lo tanto el cronograma es el siguiente:

    - Sprint 1: Semanas 1-2



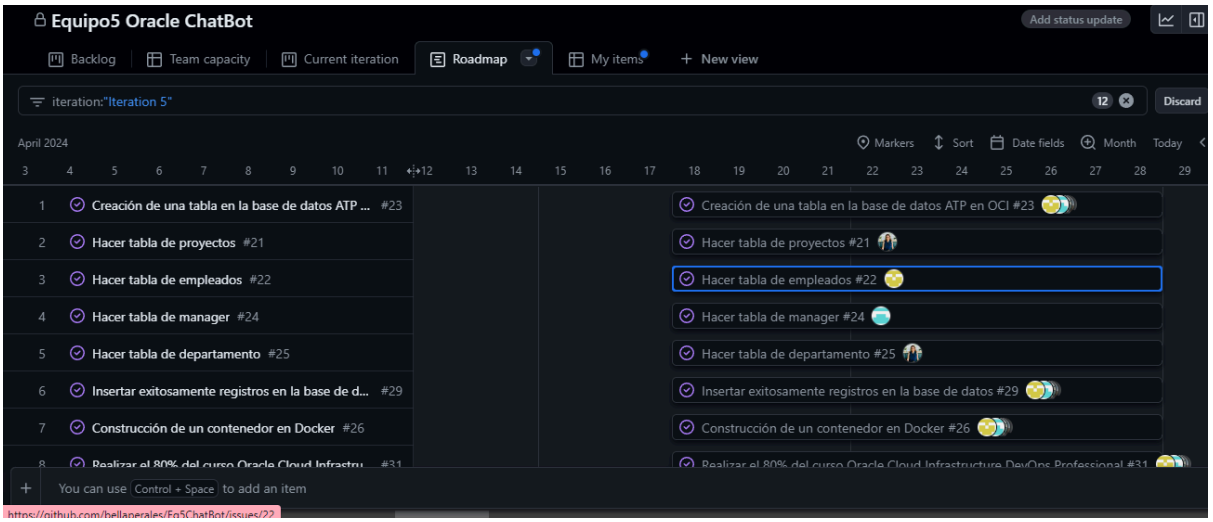
    - Sprint 2: Semanas 3-4

    - Sprint 3: Semanas 5-6



- Sprint 4: Semanas 7-8

- Sprint 5: Semanas 9-10



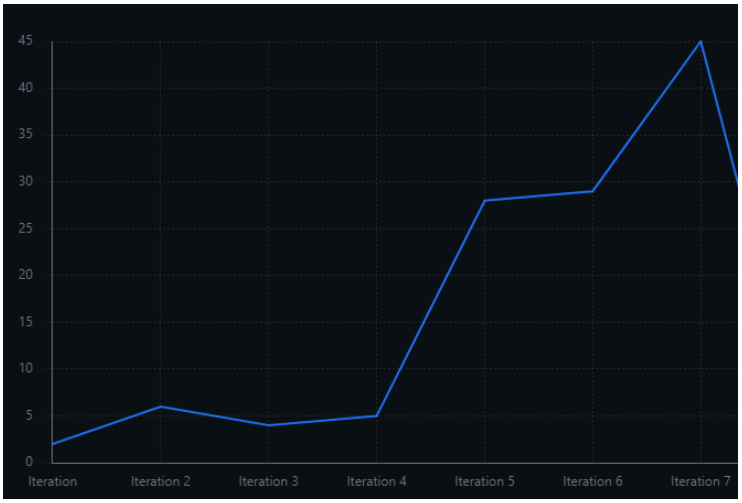
- Sprint 6: Semanas 11-12

- Sprint 7: Semanas 13-15

8. Métricas ágiles

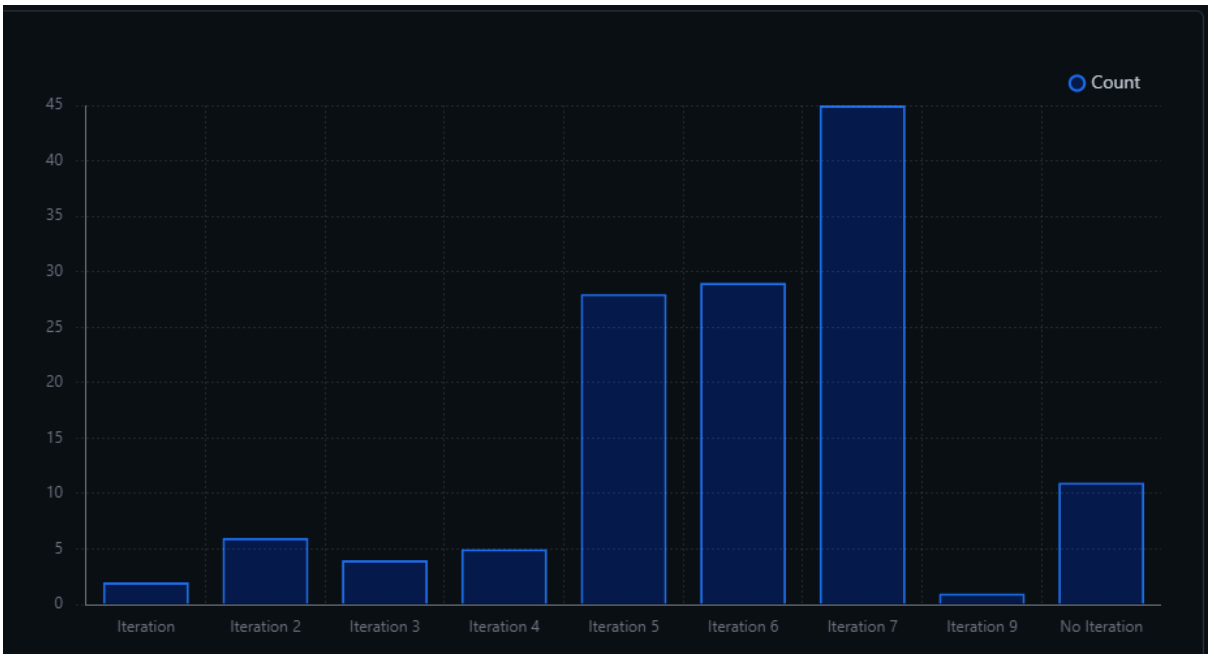
8.1 Burndown Chart

- A continuación se presenta un Burndown Chart general. Para una metodología ágil en ocasiones es conveniente tener una gráfica general para todo el proyecto seccionada por sprints. En el tiempo de trabajo hubo un total de 7 sprints, por lo que el eje horizontal denota sprints y el eje vertical las tareas que realizadas en cada sprint.



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 y el eje vertical representa el porcentaje de trabajo realizado.



-

9. Plan de Recursos

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

9.1 Presupuesto y Finanzas

El presupuesto total estimado para el proyecto es de $625,000 MXN. A continuación, se presenta la distribución detallada del presupuesto:

9.1.1 Distribución del Presupuesto:

Después de un cuidadoso análisis y estimación de costos, se ha determinado la siguiente distribución del presupuesto:

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

9.2 Recursos Humanos

Para llevar a cabo este proyecto de manera exitosa, se requiere un equipo multidisciplinario con las siguientes funciones y roles clave:

9.2.1 Roles

- Desarrolladores Java (2)
- Especialista en DevOps y CI/CD (2)
- Manager del Proyecto (1)

9.2.2 Modalidades de Trabajo

Con el fin de adaptarse a las necesidades del equipo y aprovechar las ventajas de la flexibilidad laboral, se implementarán las siguientes modalidades de trabajo:

- Remota
- Híbrida

9.3 Infraestructura y Herramientas

Para el desarrollo y despliegue del proyecto, se utilizará la siguiente 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

9.4 Desarrollo y Operaciones (DevOps)

Con el fin de agilizar el proceso de desarrollo y despliegue, se implementará un enfoque de DevOps que incluye:

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

9.5 Etapas del Proyecto

El proyecto se dividirá en las siguientes etapas principales:

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

9.6 Seguridad

La seguridad es una prioridad en este proyecto, por lo que se tomarán medidas para asegurar todos los componentes de la arquitectura de software.

9.7 Documentación y Colaboración

Para facilitar la colaboración y el seguimiento del proyecto, se utilizarán las siguientes herramientas:

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

9.8 Reserva para Imprevistos

Con el fin de estar preparados para cualquier eventualidad o ajuste no previsto, se ha asignado un fondo de reserva que representa el 10% del presupuesto total.

10. Plan de Riesgos

10.1. Matriz de Identificación de Riesgos

Esta sección presenta una lista exhaustiva de los posibles riesgos identificados para el proyecto, junto con su probabilidad, consecuencia y calificación de riesgo.

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

10.2. Matriz de Probabilidad - Impacto

Esta matriz proporciona una representación visual de los riesgos identificados en función de su probabilidad de ocurrencia y el impacto potencial en el proyecto.

		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

En esta sección, se detallan las estrategias y acciones planificadas para abordar cada uno de los riesgos identificados previamente.

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.

11. Plan de Valor Agregado

11.1. 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.

11.2. Parámetros

Para implementar el plan de valor ganado de manera efectiva, es necesario definir los parámetros iniciales que servirán como base para los cálculos y mediciones posteriores.

11.2.1 Presupuesto Total

El presupuesto total asignado para este proyecto es de $625,000.

11.2.2 SCRUM

La metodología ágil SCRUM será utilizada para la gestión del proyecto. Los detalles del enfoque SCRUM se encuentran en la sección Estrategia de Desarrollo.

11.2.3 Cronograma:

El cronograma detallado del proyecto, incluyendo las fechas estimadas para cada fase y entregable, se encuentra en la sección de Manejo de Cronograma. El tiempo total estimado para la finalización del proyecto es de 3.5 meses.

11.3. Cálculo de métricas

Las métricas clave del plan de valor ganado se calcularán tomando como base el presupuesto total del proyecto. Es importante considerar que, en un proyecto de este tipo, una parte significativa del presupuesto se destina a los sueldos del equipo, incluyendo desarrolladores y el gerente del proyecto. A continuación, se presenta la distribución estimada del presupuesto:

11.3.1 Sueldos

Equivalen al 70% del presupuesto total, dando un equivalente de $437, 500.

11.3.2 Otros costos

Equivalen al 30% del presupuesto, dividido a continuación:

- Infraestructura y herramientas de desarrollo (40%) equivale a $75,000.
- Pruebas (30%) equivalen a $56,250.
- Operaciones y mantenimiento (20%) equivale a $37,500.
- Reservas (10%) equivale a $18,750.

Las métricas clave a calcular son el planned value (PV), el earned value (EV) y el Actual Cost (AC). El PV describe el valor mensual planificado del proyecto, el EV se calcula en función del porcentaje del proyecto que se espera completar en un rango de tiempo determinado, y el AC es una estimación del costo real del proyecto en un mes específico.

- PT = $625,000
- Sueldos y equipo para los desarrolladores: 625,000 * 0.7 = $437,500
  - El equipo estará conformado por 2 desarrolladores y 1 gerente de proyecto.
  - El sueldo mensual promedio de un desarrollador Java con experiencia es de $24,000 MXN.
  - El sueldo mensual del gerente de proyecto es de $29,000 MXN, debido a su mayor nivel de responsabilidad y experiencia.

- 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
    del costo real total en el primer mes

11.4. Variaciones a considerar

Las variaciones son medidas que ayudan a determinar si el equipo podría ir sobre el presupuesto o detrás del presupuesto. Estas variaciones brindan una idea de lo que podría ocurrir en caso de que el porcentaje del proyecto o el cronograma se desvíen de lo esperado en un rango de tiempo determinado. 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.

11.5. 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.

12. Referencias

- Raeburn A. (2024) ¿Qué es un Scrum Master y cuál es su función? Recuperado de: https://asana.com/es/resources/scrum-master
- Zendesk (2023) ¿Qué es la metodología ágil y cuáles son las más utilizadas? Recuperado de: https://www.zendesk.com.mx/blog/metodologia-agil-que-es/
- Quick, L. (2024) What Is a Velocity Chart and How Do You Use It? Recuperado de: https://www.knowledgehut.com/blog/agile/velocity-chart-and-uses
- Roche J. (2024) Las 5 ceremonias Scrum: claves para la gestión de procesos. Recuperado de: https://www2.deloitte.com/es/es/pages/technology/articles/ceremonias-scrum.html
- Radigan (2024) Backlog del producto: qué es y cómo crearlo. Recuperado de: https://www.atlassian.com/es/agile/scrum/backlogs
- Arino J. (2021) Buenas prácticas de Scrum. Recuperado de: https://metodologiascrum.top/buenas-practicas-de-scrum/
- Saint-Exupery, A (2022) La matriz de Probabilidad / Impacto. Recuperado de: https://www.strategicforesight.es/blog/la-matriz-de-probabilidad-impacto/#:~:text=Una%20herramienta%20que%20se%20puede,forma%20muy%20visual%20y%20sencilla.
- rockcontent (2019) Diseño UI y UX: ¡descubre cuál es la diferencia entre ambos! Recuperado de: https://rockcontent.com/es/blog/ui-ux/
- Landau P. (2023) Cost Performance Index (CPI) In Project Management. Recuperado de: https://www.projectmanager.com/blog/cost-performance-index
- Yarbrough Q. (2021) Schedule Performance Index (SPI): An Introduction. Recuperado de: https://www.projectmanager.com/blog/schedule-performance-index-spi