ACTIVIDAD 2 - DOCUMENTO DE FORMULACIÓN DEL PROYECTO.

PROYECTO DE SOFTWARE

UNIDAD DE APRENDIZAJE 1

TATIANA CABRERA

JULIAN CAMILO ROJAS SASTOQUE

UNIVERSIDAD IBEROAMERICANA

FACULTAD DE INGENIERIA

4/10/2025

Contextualización de la necesidad

En el entorno actual del desarrollo de software, los equipos enfrentan múltiples retos relacionados con la gestión eficiente de proyectos, la seguridad de credenciales y la integración de herramientas externas. Muchos equipos utilizan distintas plataformas para cada necesidad: una para gestionar tareas (Trello, Jira), otra para almacenar código (GitHub), y otras para ejecutar contenedores o pruebas (Docker).

Esto genera fragmentación, pérdida de trazabilidad y dificultades en la coordinación del equipo. Por lo tanto, se identifica la necesidad de contar con un sistema unificado que centralice la planificación, ejecución y supervisión técnica de proyectos de software, integrando los servicios más utilizados por los desarrolladores.

Planteamiento del problema

Los equipos de desarrollo suelen invertir tiempo en cambiar entre distintas herramientas para administrar tareas, revisar repositorios o desplegar contenedores. Esto no solo aumenta la complejidad operativa, sino que también eleva los riesgos de errores humanos, duplicación de información y fugas de seguridad (por el manejo manual de credenciales o tokens).

Por ello, se requiere una solución centralizada que permita gestionar proyectos, tareas, repositorios y contenedores Docker desde una misma plataforma, con autenticación segura y funciones colaborativas.

Alcance del proyecto

Alcance general

El proyecto abarcará el diseño y desarrollo de una plataforma web que permita a los usuarios registrar proyectos, asignar tareas, monitorear su progreso, y gestionar recursos tecnológicos asociados como repositorios GitHub y contenedores Docker.

Alcance específico

- Implementación de un módulo de gestión de proyectos y tareas, con trazabilidad por etapas.
- Integración con la API de GitHub para visualizar repositorios, ramas y commits.
- Integración con Docker Engine API para gestionar contenedores desde la interfaz.
- Módulo de almacenamiento seguro de credenciales con cifrado.
- Generación y almacenamiento de códigos OTP para autenticación en dos pasos.
- Panel de control con indicadores de avance del proyecto.
- Acceso mediante autenticación de usuario y roles definidos (administrador, desarrollador, observador).

Restricciones

- El sistema inicial se enfocará en entornos locales o de prueba (no producción empresarial).
- La integración con Docker se limitará a funciones básicas: creación, arranque y detención de contenedores.
- Se requerirá conexión a Internet para sincronización con GitHub.

Criterios de aceptación

- El sistema debe permitir registrar al menos 3 proyectos y 10 tareas por proyecto.
- Debe conectarse exitosamente con una cuenta de GitHub para listar repositorios.
- Debe poder crear y detener contenedores desde la interfaz.
- Las credenciales deben cifrarse correctamente.

Estructura del Desglose del Trabajo (EDT)

Nivel	Entregable principal	Subtareas o componentes
1	Análisis y planificación del proyecto	Identificación de requerimientos, análisis de riesgos, selección de tecnología
2	Diseño del sistema	Diagramas UML, diseño de base de datos, arquitectura del sistema
3	Desarrollo de módulos	Gestión de proyectos, integración GitHub, gestión Docker, credenciales seguras
4	Pruebas y validación	Pruebas unitarias, pruebas de integración, validación con usuarios
5	Documentación y entrega final	Manual de usuario, documentación técnica, repositorio y tablero ágil

Objetivos

Objetivo general:

Desarrollar un sistema integral que permita la gestión de proyectos de software, integrando funcionalidades para la administración de tareas, repositorios GitHub, contenedores Docker y credenciales seguras.

Objetivos específicos:

1. Analizar las necesidades de gestión de proyectos tecnológicos.
2. Diseñar la arquitectura del sistema y su base de datos.
3. Implementar un módulo de gestión de tareas y seguimiento de etapas.
4. Desarrollar la integración con la API de GitHub para sincronizar repositorios.
5. Implementar la conexión con Docker para crear y administrar contenedores.
6. Garantizar la seguridad de la información mediante cifrado y autenticación OTP.
7. Evaluar el funcionamiento del sistema mediante pruebas y retroalimentación de usuarios.

Metodología ágil seleccionada: SCRUM

Se utilizará la metodología Scrum por su enfoque iterativo e incremental, que permite gestionar cambios y priorizar funcionalidades durante el desarrollo.

- Roles: Product Owner, Scrum Master y Equipo de Desarrollo.
- Artefactos: Backlog del producto, sprint backlog, tablero de tarea.
- Enlace de tablero: https://trello.com/b/3ipHLcE7/devi

Justificación

Corto plazo: Crear una herramienta funcional para gestionar proyectos de software en entornos académicos o de pequeña escala.

Mediano plazo: Integrar el sistema con APIs de terceros (GitHub, Docker) para facilitar la automatización del flujo de desarrollo.

Largo plazo: Ampliar la plataforma a un entorno colaborativo en la nube con monitoreo de rendimiento y despliegue continuo.

Mapa y clasificación de Stakeholders

Stakeholder	Rol / Interés	Clasificación
Equipo de desarrollo	Implementación técnica del sistema	Interno / Alto interés
Docente / Evaluador	Supervisión y evaluación académica	Externo / Alto poder
Usuarios finales (desarrolladores)	Uso práctico de la plataforma	Externo / Alto interés
Administrador del sistema	Gestión de seguridad y despliegue	Interno / Alto poder

Matriz de riesgos

Riesgo	Probabilidad	Impacto	Estrategia de mitigación
Fallo en la conexión con API de GitHub	Media	Alta	Implementar manejo de errores y reintentos automáticos
Fuga de credenciales	Baja	Muy alta	Uso de cifrado AES y tokens encriptados
Retraso en el desarrollo	Alta	Media	Dividir en sprints cortos con entregas parciales
Errores en la integración Docker	Media	Media	Probar en entornos controlados y documentar
Pérdida de datos	Baja	Alta	Respaldos automáticos en base de datos

Cronograma (resumen por semanas)

Semana	Actividad
1	Análisis de requerimientos y diseño general
2	Diseño de base de datos y arquitectura
3-4	Desarrollo módulo de proyectos y tareas
5-6	Integración con GitHub y Docker
7	Pruebas, documentación y presentación

Presupuesto estimado

Recurso	Descripción	Costo estimado (USD)
Hosting y dominio	Para pruebas en la nube	30
Servicios GitHub API / Docker	Gratuitos (nivel básico)	0
Herramientas de desarrollo (VSCode, Postman)	Gratuitas	0
Tiempo de desarrollo (100 horas)	Valor hora promedio 10 USD	1000
Total aproximado		1030 USD

Diagrama de flujo:

A continuación el enlace para visualizar el diagrama de flujo del sistema Devi realizado en Miro:

https://miro.com/app/board/uXjVJ-kvv3Y=/?share_link_id=418529066557

Historias de Usuario

Módulo de autenticación y credenciales

HU-01: Registro de usuario

- **Como** nuevo usuario
- **Quiero** poder registrarme en la plataforma Devi
- **Para** acceder a mis proyectos y funcionalidades de gestión.
  Criterios de aceptación:
- Se deben solicitar nombre, correo y contraseña.
- La contraseña debe almacenarse cifrada.
- El sistema debe validar si el correo ya existe.
  Prioridad: Alta
  Estimación: 2 puntos

HU-02: Inicio de sesión con OTP

- **Como** usuario registrado
- **Quiero** iniciar sesión mediante correo, contraseña y código OTP
- **Para** reforzar la seguridad de mi cuenta.
  Criterios de aceptación:
- El sistema debe enviar un código OTP de 6 dígitos al correo.
- El OTP debe tener una duración máxima de 5 minutos.
- Si el OTP es incorrecto o expira, el usuario no debe poder ingresar.
  Prioridad: Alta
  Estimación: 3 puntos

HU-03: Gestión de credenciales seguras

- **Como** usuario
- **Quiero** guardar mis credenciales de servicios externos (GitHub, Docker, etc.)
- **Para** no tener que autenticarlas cada vez que use una integración.
  Criterios de aceptación:
- Las credenciales deben cifrarse antes de almacenarse.
- Solo el usuario propietario puede acceder a sus credenciales.
- Las credenciales deben poder actualizarse o eliminarse.
  Prioridad: Alta

Módulo de gestión de proyectos

HU-04: Crear proyecto

- **Como** usuario
- **Quiero** crear un nuevo proyecto con nombre, descripción y fechas
- **Para** gestionar tareas y recursos asociados.
  Criterios de aceptación:
- El sistema debe validar los campos obligatorios.
- El proyecto se debe guardar en la base de datos.
- El usuario debe recibir confirmación del registro.
  Prioridad: Alta

HU-05: Editar o eliminar proyecto

- **Como** usuario
- **Quiero** poder actualizar o eliminar un proyecto existente
- **Para** mantener la información actualizada.
  Criterios de aceptación:
- El sistema debe permitir editar nombre, descripción y fechas.
- El sistema debe solicitar confirmación antes de eliminar.
- Las tareas asociadas deben gestionarse correctamente al eliminar.
  Prioridad: Media

Módulo de tareas y seguimiento

HU-06: Crear y gestionar tareas

- **Como** usuario del proyecto
- **Quiero** crear tareas y asignarlas a miembros del equipo
- **Para** organizar el trabajo y hacer seguimiento del avance.
  Criterios de aceptación:
- Cada tarea debe tener nombre, descripción, responsable, fecha límite y estado.
- Se debe poder cambiar el estado (Por hacer, En progreso, Hecho).
- El sistema debe calcular el progreso total del proyecto.
  Prioridad: Alta

HU-07: Visualizar tareas en tablero Kanban

- **Como** usuario
- **Quiero** ver mis tareas en un tablero visual
- **Para** identificar fácilmente el avance y estado de cada actividad.
  Criterios de aceptación:
- El tablero debe tener tres columnas (Por hacer, En progreso, Hecho).
- Debe permitir arrastrar tareas entre columnas.
- El progreso debe actualizarse automáticamente.
  Prioridad: Media

Integración con GitHub

HU-08: Conectar con cuenta de GitHub

- **Como** usuario
- **Quiero** conectar mi cuenta de GitHub mediante OAuth
- **Para** acceder a mis repositorios directamente desde Devi.
  Criterios de aceptación:
- El sistema debe redirigir a la autorización de GitHub.
- Al aprobar, se debe obtener y guardar el token de acceso.
- Debe mostrarse una lista de repositorios disponibles.
  Prioridad: Alta

HU-09: Asociar repositorio a proyecto

- **Como** usuario
- **Quiero** vincular un repositorio de GitHub a un proyecto en Devi
- **Para** visualizar commits, ramas y versiones.
  Criterios de aceptación:
- El sistema debe listar los repositorios del usuario autenticado.
- El usuario debe poder seleccionar uno y asociarlo al proyecto.
- Se debe mostrar información básica del repositorio (commits, branch).
  Prioridad: Media

Integración con Docker

HU-10: Listar contenedores Docker

- **Como** usuario
- **Quiero** visualizar los contenedores activos, detenidos y en error
- **Para** monitorear el estado de mis entornos.
  Criterios de aceptación:
- El sistema debe listar nombre, imagen, estado y puertos del contenedor.
- Los datos deben actualizarse en tiempo real o al refrescar.
  Prioridad: Media

HU-11: Gestionar contenedores Docker

- **Como** usuario
- **Quiero** poder crear, iniciar o detener contenedores desde Devi
- **Para** controlar mis entornos de desarrollo sin usar la consola.
  Criterios de aceptación:
- El sistema debe ofrecer botones de acción (iniciar/detener/eliminar).
- Debe mostrar mensajes de éxito o error.
- Los cambios deben reflejarse inmediatamente en la lista.
  Prioridad: Media

Módulo de métricas y reportes

HU-12: Visualizar métricas de proyectos

- **Como** usuario
- **Quiero** ver estadísticas del progreso, tareas completadas y tiempo restante
- **Para** evaluar el avance general del proyecto.
  Criterios de aceptación:
- Se deben mostrar porcentajes, gráficos y fechas.
- La información debe actualizarse en tiempo real según las tareas.
  Prioridad: Media

Módulo de seguridad y cierre

HU-13: Cerrar sesión

- **Como** usuario
- **Quiero** cerrar sesión de forma segura
- **Para** proteger mis credenciales y datos.
  Criterios de aceptación:
- El sistema debe invalidar la sesión actual.
- Debe redirigir al inicio.
- No debe permitir volver atrás sin autenticación.
  Prioridad: Alta

Requisitos funcionales y no funcionales

A continuación, se presentan los requisitos definidos para el sistema Devi, los cuales fueron levantados a partir del análisis de la necesidad y los objetivos del proyecto.

Requisitos funcionales

ID	Nombre del requisito	Descripción	Criterios de aceptación
RQF-01	Registro de usuarios	El sistema debe permitir registrar nuevos usuarios con nombre, correo electrónico y contraseña cifrada.	El sistema valida campos obligatorios y evita correos duplicados.
RQF-02	Inicio de sesión con OTP	El usuario debe poder autenticarse ingresando sus credenciales y un código OTP enviado al correo.	El OTP debe tener una vigencia máxima de 5 minutos.
RQF-03	Gestión de credenciales seguras	El sistema debe permitir almacenar de forma cifrada las credenciales de servicios externos (GitHub, Docker, etc.).	Solo el usuario autenticado puede acceder a sus credenciales.
RQF-04	Creación y edición de proyectos	El usuario podrá crear, editar y eliminar proyectos con nombre, descripción y fechas.	Los proyectos deben almacenarse en base de datos y confirmar su creación.
RQF-05	Gestión de tareas	El usuario podrá crear tareas, asignarlas, definir fechas límite y estados de avance.	Las tareas deben mostrarse en un tablero Kanban con sus respectivos estados.
RQF-06	Seguimiento del progreso	El sistema calculará y mostrará el porcentaje de avance del proyecto según las tareas completadas.	El progreso se actualiza automáticamente al modificar tareas.
RQF-07	Integración con GitHub	El sistema debe conectarse mediante OAuth con GitHub y mostrar los repositorios del usuario.	Se deben mostrar los repositorios disponibles tras la autenticación exitosa.
RQF-08	Asociación de repositorios	Permite vincular un repositorio de GitHub a un proyecto existente en Devi.	Se muestra información del repositorio seleccionado (commits, ramas).
RQF-09	Listar contenedores Docker	El sistema debe mostrar los contenedores existentes (activos, detenidos, error).	Debe mostrar nombre, imagen, estado y puertos del contenedor.
RQF-10	Control de contenedores Docker	El sistema debe permitir crear, iniciar o detener contenedores Docker desde la interfaz.	Se deben mostrar mensajes de éxito o error según la acción.

RQF-11	Visualización de métricas	El usuario podrá consultar estadísticas y gráficos del avance de los proyectos.	Se deben mostrar porcentajes, tareas completadas y fechas límite.
RQF-12	Cierre de sesión	El usuario debe poder cerrar sesión de forma segura.	La sesión se invalida y el sistema redirige al inicio.

Requisitos no funcionales

ID	Nombre del requisito	Descripción	Criterios de aceptación
RQNF-01	Seguridad de datos	Toda la información sensible (contraseñas, tokens, credenciales) debe almacenarse cifrada.	Se usa cifrado AES-256 o equivalente.
RQNF-02	Disponibilidad	El sistema debe estar disponible el 99% del tiempo para los usuarios registrados.	El tiempo de inactividad mensual no debe superar el 1%.
RQNF-03	Rendimiento	Las consultas y operaciones no deben superar los 3 segundos de respuesta.	Se mide con herramientas de rendimiento del servidor.
RQNF-04	Escalabilidad	El sistema debe poder soportar múltiples usuarios concurrentes sin degradación significativa.	Pruebas con al menos 50 usuarios concurrentes mantienen rendimiento aceptable.
RQNF-05	Usabilidad	La interfaz debe ser intuitiva, moderna y fácil de usar para usuarios sin conocimientos técnicos avanzados.	Pruebas con usuarios deben arrojar una satisfacción ≥ 80%.
RQNF-06	Compatibilidad	El sistema debe ser accesible desde navegadores modernos (Chrome, Firefox, Edge).	Debe funcionar correctamente en versiones actuales de cada navegador.
RQNF-07	Mantenibilidad	El código debe seguir estándares de buenas prácticas y permitir la fácil incorporación de nuevas funcionalidades.	La documentación técnica y comentarios deben estar actualizados.
RQNF-08	Confiabilidad	El sistema debe manejar errores sin pérdida de datos y registrar eventos críticos.	Implementación de logs y manejo de excepciones.
RQNF-09	Portabilidad	El sistema debe poder ejecutarse en diferentes entornos (local, nube o contenedores Docker).	Debe existir una guía de despliegue con Docker Compose.

RQNF-10	Cumplimiento normativo	El sistema debe cumplir con las políticas de privacidad y normas de protección de datos.	Cumplimiento con RGPD o norma equivalente.

Conclusiones

El desarrollo del sistema Devi busca ofrecer una herramienta innovadora que centralice la gestión de proyectos de software con integración directa a las herramientas más utilizadas por los desarrolladores. Su enfoque en la automatización, la seguridad y la trazabilidad permitirá optimizar los procesos de trabajo y mejorar la eficiencia de los equipos técnicos.

Referencias

- **Pressman, Roger S., (2021) Ingeniería de software. McGraw-Hill Interamericana.** Capitulo 24, 25, 26 página de la 490 a 548
- **Tablero Kanban:** https://trello.com/b/3ipHLcE7/devi
- **Repositorio de código: https://github.com/JulianRojasS/devi**
- Diagrama de flujo Devi: https://miro.com/app/board/uXjVJ-kvv3Y=/?share_link_id=418529066557
- Omaña, M. (2012). Manufactura esbelta: una contribución para el desarrollo de software con calidad. Red Enlace página de la 14 a 18.