REPUBLICA BOLIVARIANA DE VENEZUELA

UNIVERSIDAD NACIONAL EXPERIMENTAL PARA LAS

TELECOMUNICACION E INFORMATICA

VICERRECTORADO ACADEMICO

PROGRAMA NACIONAL DE FORMACION EN INGENIERIA EN

INFORMATICA

Sistema de gestion de actividades con IA WorkLyst

Tutora:

Prof. Yuly Delgado

Autores:

Br. Pedro Castro V-32.736.571

Br. Orlando Lopez V-31.332.501

Br. Jean Diaz V-30.831.319

Br. Wrallean Brito V-24.774.283

Caracas, 12 de diciembre del 2025

Indice General

Introduccion

La gestión eficiente de proyectos colaborativos es un pilar fundamental en el ámbito académico y de desarrollo tecnológico, especialmente en un entorno de alta demanda como el de la Universidad Nacional Experimental para las Telecomunicaciones e Informática (UNETI). A menudo, la coordinación de equipos multidisciplinarios y la subdivisión de tareas complejas generan fricciones que obstaculizan la productividad y la trazabilidad del trabajo.

Este documento presenta el "Sistema de gestión de actividades con IA WorkLyst" , una plataforma web diseñada como una solución innovadora para optimizar la colaboración entre universitarios y desarrolladores en la ejecución de proyectos complejos.

El propósito principal de WorkLyst es superar las limitaciones de las herramientas de gestión tradicionales, integrando un potente agente de Inteligencia Artificial (IA) especializado. Esta integración no solo facilita la organización de tareas y mejora la comunicación entre equipos, sino que también automatiza procesos clave, como la asignación inteligente de tareas con base en las habilidades y disponibilidad de los usuarios, y ofrece herramientas inteligentes para agilizar el flujo de trabajo.

A lo largo de este informe, se detallará el propósito, la justificación y el alcance del proyecto. Posteriormente, se expondrán los objetivos funcionales, la identificación de los actores clave y la arquitectura técnica que soporta la escalabilidad y el enfoque de IA de la plataforma. Finalmente, se abordarán los procesos de negocio, los requisitos de seguridad y el plan de

desarrollo y pruebas para la implementación exitosa de WorkLyst en el contexto universitario.

Proposito del Proyecto

El propósito de esta aplicación es facilitar la colaboración entre universitarios y desarrolladores en la ejecución de proyectos complejos, mediante el uso de inteligencia artificial. La plataforma busca optimizar la subdivisión de tareas, mejorar la comunicación entre equipos y ofrecer herramientas inteligentes que agilicen el flujo de trabajo.

Justificación

Este proyecto representa una oportunidad para demostrar nuestras capacidades técnicas y de diseño, al tiempo que propone una solución innovadora en el ámbito de la gestión colaborativa. A diferencia de herramientas tradicionales como Trello, esta aplicación incorpora un agente de IA especializado, chat grupal integrado y funcionalidades avanzadas que potencian la productividad y la organización de los equipos.

Alcance General

La aplicación permitirá:

- Crear proyectos personalizados.
- Formar equipos y añadir miembros.
- Generar tableros tipo Kanban o con otras plantillas visuales.
- Interactuar mediante chat grupal y chat con IA.
- Asignar tareas y roles de forma inteligente.
- Utilizar drag & drop para organizar tareas.

Actualmente no se contemplan restricciones técnicas ni de funcionalidad, lo que permite una evolución flexible del sistema.

Usuarios Directos

Los usuarios principales serán estudiantes de la UNETI, con proyección a desarrolladores de otras instituciones y comunidades. Aunque la app puede ser utilizada por cualquier persona interesada en la gestión de proyectos, su enfoque está orientado a perfiles técnicos y académicos.

Servicios Clave

- Creación y gestión de proyectos
- Inclusión de compañeros y asignación de roles
- Chat grupal e interacción con IA
- Organización de tareas mediante tableros visuales
- Asignación inteligente de tareas

Ambito de Operacion

La aplicación será una plataforma web responsiva, accesible desde cualquier dispositivo con conexión a internet. No se contempla operación offline en esta primera etapa.

Contexto Universitario

El sistema será implementado en el entorno académico de la Universidad Nacional Experimental de las Telecomunicaciones e Informática (UNETI), donde existe un alto flujo de actividades colaborativas. Se busca dar soporte a los procesos de gestión de proyectos, facilitando la coordinación entre equipos, el seguimiento de avances y la visualización clara de tareas.

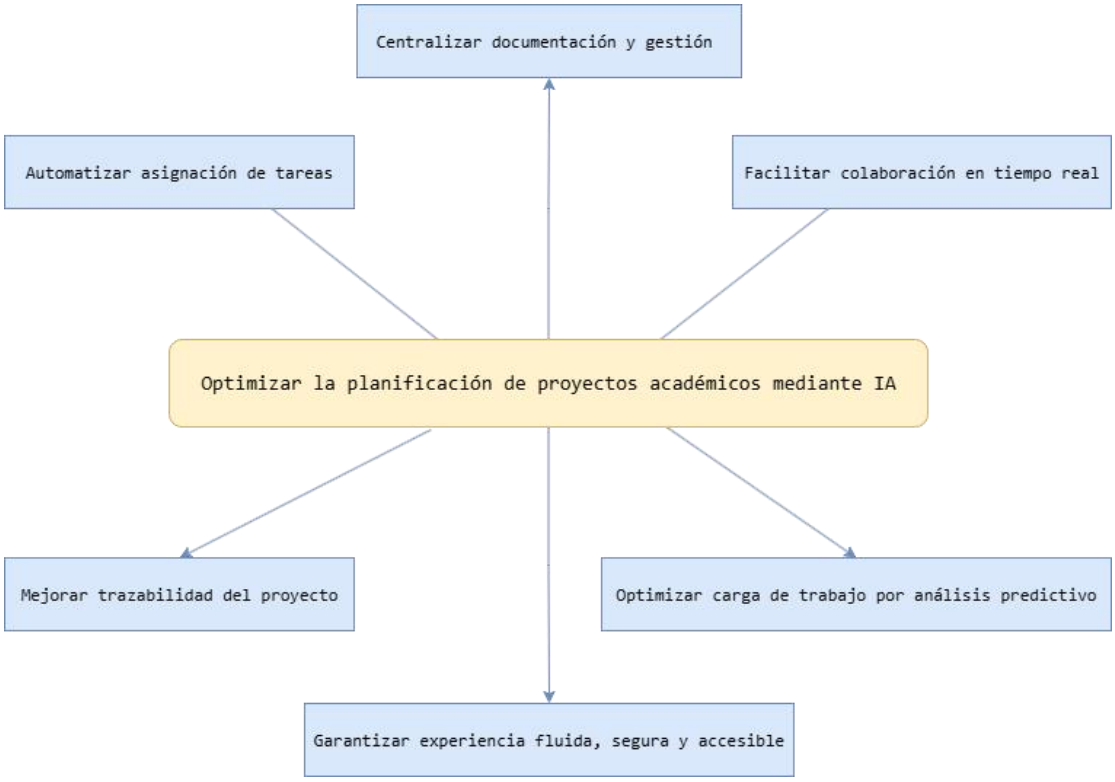
El sistema propone una interfaz accesible y moderna para mejorar la distribución de responsabilidades, reducir tiempos en la planificación manual y mejorar la experiencia del usuario mediante la integración de herramientas interactivas basadas en IA.

Objetivos del Negocio

1. **Automatizar la asignación de tareas** mediante inteligencia artificial, analizando habilidades, carga de trabajo y disponibilidad de los usuarios.
2. **Facilitar la colaboración en tiempo real** entre estudiantes y docentes, mediante herramientas de mensajería, menciones, comentarios y notificaciones.
3. **Centralizar la gestión de proyectos académicos** en una interfaz intuitiva que permita visualizar tareas, fechas límite, dependencias y prioridades.
4. **Mejorar la trazabilidad de carga del trabajo** registrando el historial de cambios, actividades realizadas y comentarios relevantes durante el ciclo de vida del proyecto.
5. **Optimizar la distribución de carga de trabajo** con análisis predictivo y sugerencias automáticas de reasignación.
6. **Brindar una experiencia fluida y segura**, compatible con múltiples dispositivos, mediante una interfaz responsiva, accesible y bien organizada.
7. **Reducir la dependencia de planificación manual**, mediante tableros configurables tipo Kanban, uso de plantillas predefinidas y formularios inteligentes.

Diagrama de Objetivos del Sistema

Se adjunta a continuación el diagrama jerárquico que representa los objetivos generales y específicos del sistema.



Identificación de los actores

Una parte importante de nuestro proyecto son sus actores a continuación se presenta una lista de ellos:

- Administrador del Sistema
- Líder del Proyecto
- Usuario regular
- Colaborador
- Visitante

Ahora que sabemos que actores se relacionan con nuestro proyecto podemos empezar a detallar las funcionalidades y procesos relacionados con cada uno.

Administrador del Sistema

Este rol es el más importante dentro del proyecto, ya que tiene responsabilidades críticas y funcionalidades clave como: Crear, modificar y eliminar roles; otorgar permisos globales dentro del sistema; gestionar usuarios; validar credenciales de acceso; supervisar el funcionamiento general y la estabilidad del sistema; resolver conflictos de edición simultánea; controlar sesiones activas; y garantizar un rendimiento óptimo y seguro del sistema en todo momento.

Líder del Proyecto

Este rol ocupa el segundo nivel en la jerarquía del sistema y maneja funciones esenciales para la gestión efectiva del proyecto. Sus funcionalidades incluyen: crear y configurar proyectos desde cero; definir objetivos SMART (específicos, medibles, alcanzables, relevantes y temporales); utilizar planillas predefinidas que facilitan la organización; configurar tableros de trabajo como KANBAN o SCRUM para el seguimiento ágil; gestionar y asignar colaboradores dentro de los proyectos; establecer fechas límite claras; clasificar tareas según prioridad y dificultad; establecer dependencias entre tareas; supervisar y aprobar cambios realizados; y comentar tareas para mejorar la comunicación.

Usuario regular

Este es el rol predeterminado asignado a todos los usuarios registrados en el sistema. Sus funciones principales incluyen: registro y autenticación segura de usuarios; edición y actualización de su perfil personal; completar el formulario inicial para personalizar su experiencia; gestionar su propia sección o área de trabajo; visualizar las tareas que le han sido asignadas; comentar en las tareas para aportar información o dudas; desplazar tareas entre diferentes tableros según el avance; adjuntar archivos relevantes; mencionar a otros colaboradores para llamar su atención; y consultar el historial completo de actividades realizadas.

Colaborador

Este actor es una extensión del usuario regular, que adquiere nuevas funcionalidades una vez es invitado a participar en un proyecto específico dentro del sistema. Entre sus nuevas capacidades están: visualizar las tareas asignadas dentro del proyecto; cambiar el estado o progreso de las tareas; adjuntar archivos relacionados; usar menciones para comunicarse con otros miembros; recibir notificaciones en tiempo real sobre actividades de otros usuarios en el proyecto; y consultar el historial detallado del proyecto para mantenerse informado.

Visitante

Este actor representa a los usuarios que no están registrados ni han iniciado sesión en el sistema. Su acceso es muy limitado y sus funcionalidades se restringen a: visualizar contenido público disponible; consultar noticias o reglamentos institucionales; y registrarse para obtener acceso completo al sistema.

Organigrama de Actores

Teniendo presente los actores relacionados con el sistema del proyecto, podemos presentar su organigrama para visualizar claramente la estructura y jerarquía de roles dentro del sistema.

Objetivos del Sistema y sus Atributos Relevantes

1. Objetivos de Gestión de Usuarios

Usuario

Atributos:

- ID
- Nombre
- Correo electrónico
- Contraseña (encriptada)
- Rol
- Disponibilidad (horarios/carga laboral actual)
- Historial de sesiones (fechas de inicio/cierre)

Rol

Atributos:

- ID
- Nombre
- Permisos

Sesion

Atributos:

- ID
- Usuario_ID
- Fecha de inicio
- Fecha de cierre
- Estado (activa/inactiva)

2. Objetos de Gestion de Proyectos

Proyecto

Atributos:

- ID
- Nombre
- Descripción
- Fecha de inicio
- Fecha
- límite
- Objetivos SMART
- Estado
- Plantilla utilizada
- Historial de cambios

Tablero

Atributos:

- ID
- Proyecto_ID
- Tipo
- Columnas

Plantilla

Atributos:

- ID
- Nombre
- Estructura predefinida

3. Objetivos de Gestion de Tareas

Tarea

Atributos:

- ID
- Proyecto_ID
- Título
- Descripción
- Prioridad
- Dificultad
- Fecha límite
- Estado
- Usuario asignado
- Dependencias
- Comentarios

Comentarios

Atributos:

- ID
- Tarea_ID
- Usuario_ID
- Contenido
- Fecha de creación
- Menciones

4. Objetivos de Inteligencia Artificial

Perfil de Habilidades

Atributos:

- Usuario_ID
- Habilidades
- Nivel de competencia
- Historial de tareas completadas

Chatbot de IA

Atributos:

- ID de conversación
- Usuario_ID
- Historial de mensajes
- Comandos soportados

5. Objetivos de Colaboración

Notificación

Atributos:

- ID
- Usuario destino
- Tipo
- Contenido
- Fecha
- Estado

Archivo Adjuntos

Atributos:

- ID
- Tarea_ID o Proyecto_ID
- Nombre
- Tipo
- Tamaño
- Fecha de subida

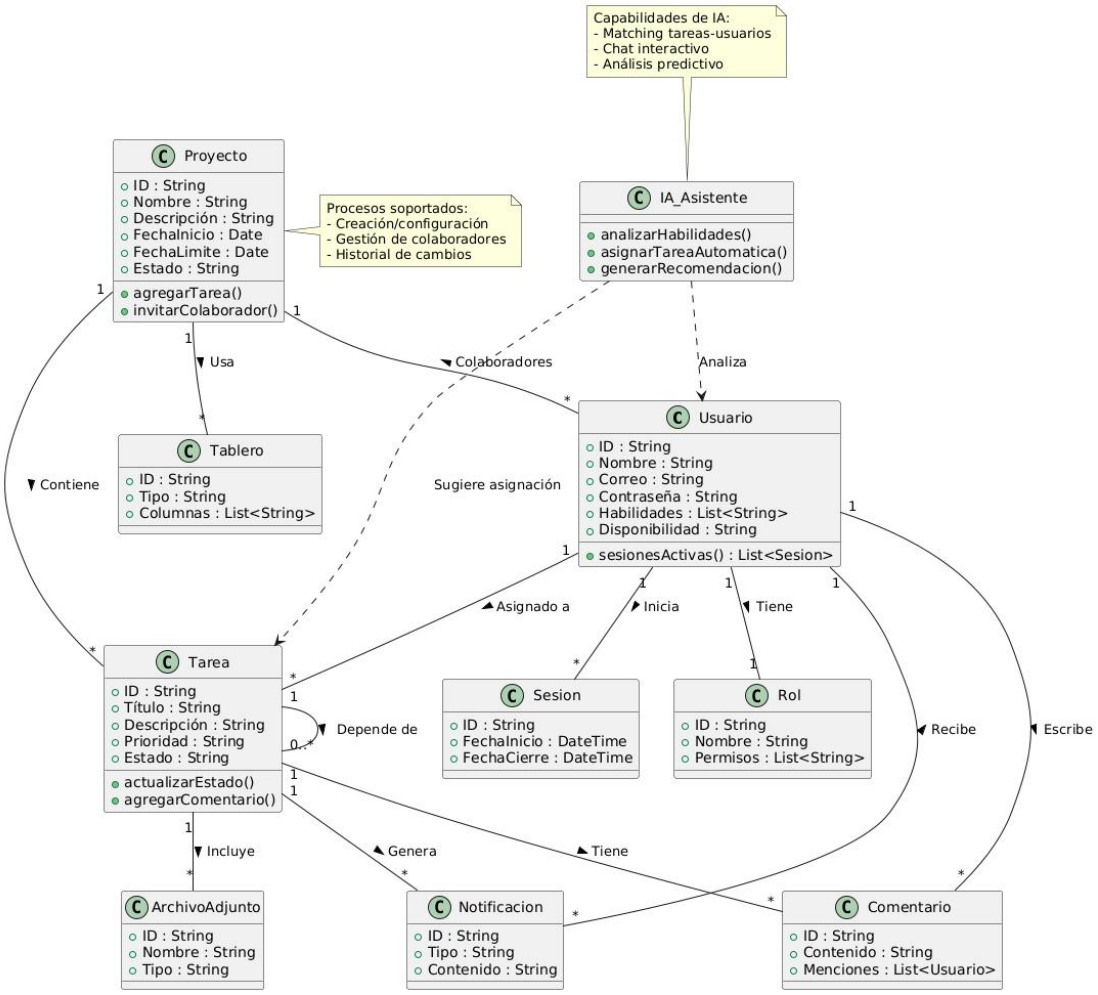
6. Objetivos de Auditoria

Historial de Cambios

Atributos:

- ID
- Proyecto_ID o Tarea_ID
- Usuario_ID
- Acción
- Fecha
- Detalles

Diagrama de Clases



Matriz Procesos vs Objetivos/Clases

Proceso de Negocio	Clases/Objetos Involucrados	Tipo de Relación
Gestión de Usuarios		
Registro/Autenticación	Usuario, Rol, Sesion	El objeto Usuario se asocia con Rol y registra Sesion durante autenticación.
Gestión de Perfiles	Usuario, PerfilHabilidades (implícito en Usuario)	Usuario almacena atributos de perfil (habilidades, disponibilidad).
Gestión de Roles	Rol, Usuario	Rol define permisos asignados a cada Usuario.
Gestión de Proyectos		
Creación/Configuración	Proyecto, Tablero, Plantilla	Proyecto usa Tablero y puede basarse en Plantilla.
Invitación de Colaboradores	Proyecto, Usuario	Relación muchos-a-muchos: Proyecto tiene múltiples Usuario (colaboradores).
Historial de Cambios	Proyecto, HistorialCambios	Proyecto registra modificaciones en HistorialCambios.
Gestión de Tareas		
Creación/Asignación	Tarea, Usuario, Proyecto	Tarea pertenece a un Proyecto y se asigna a Usuario.
Dependencias entre Tareas	Tarea (autorrelación)	Una Tarea puede depender de otras mediante relación recursiva.

Comentarios y Archivos	Tarea, Comentario, ArchivoAdjunto	Tarea agrega Comentario y ArchivoAdjunto.
Procesamiento de IA		
Matching Tareas-Usuarios	IA_Asistente, Usuario, Tarea	IA_Asistente analiza Usuario y sugiere asignación a Tarea.
Chat Interactivo	IA_Asistente, Usuario, Comentario	El chat usa Comentario para interacción usuario-IA.
Análisis Predictivo	IA_Asistente, Proyecto, Tarea	IA analiza datos históricos de Proyecto y Tarea.
Gestión de Colaboración		
Notificaciones	Notificacion, Usuario, Tarea	Tarea genera Notificacion para Usuario.
Sincronización en Tiempo Real	Proyecto, Tarea, Tablero	Cambios en Tarea actualizan Tablero y Proyecto en tiempo real.

Procesos del Negocio y Apoyo de la Aplicación

El proyecto busca desarrollar una aplicación web para la planificación de actividades de usuarios y desarrolladores con soporte de inteligencia artificial. Esta aplicación surge de la necesidad de los estudiantes y desarrolladores de la UNETI de contar con herramientas automatizadas para la asignación de tareas según las habilidades de cada integrante, superando las limitaciones de las plataformas actuales. La solución propuesta integrará un chat interactivo con IA para optimizar la organización y distribución de proyectos académicos y tecnológicos.

Identificación y Descripción de Procesos de Negocio.

Los procesos de negocio se han clasificado en las siguientes áreas principales, siguiendo la estructura de la cadena de valor y los requisitos funcionales:

1. Gestion de Usuarios

- **Registro y Autenticación de Usuarios:** Proceso mediante el cual los usuarios se registran por primera vez en el sistema y luego acceden a él con sus credenciales.
- **Gestión de Perfiles:** Proceso donde los usuarios crean, editan y actualizan su información personal y de contacto.
- **Gestión de Roles y Permisos:** Proceso de asignación de roles predefinidos a los usuarios y la gestión de las acciones que pueden realizar según su rol.
- **Recolección de Datos Iniciales:** Proceso de recopilación de información clave del usuario (habilidades, intereses) después del registro.
- **Gestión de Sesiones:** Proceso de control del inicio, mantenimiento y cierre de las sesiones de usuario.
- **Validación de Credenciales:** Verificación de la exactitud de los datos de acceso durante el registro y la autenticación.

2. Gestion de Proyectos

- **Creación y Configuración de Proyectos:** Proceso de inicio de un nuevo proyecto, incluyendo la definición de nombre, descripción y fechas.
- **Definición de Objetivos y Alcance:** Proceso de establecer los objetivos SMART y documentar los entregables y exclusiones del proyecto.
- **Configuración de Tableros:** Proceso de selección y personalización de diferentes tipos de tableros (Kanban, Scrum) para la visualización de tareas.
- **Uso de Plantillas Predefinidas:** Proceso de selección y aplicación de plantillas existentes para estructurar nuevos proyectos.
- **Gestión de Colaboradores y Permisos por Proyecto:** Proceso de invitación de usuarios a un proyecto y asignación de roles y permisos específicos dentro del mismo.
- **Registro y Visualización del Historial de Cambios del Proyecto:** Proceso de mantener una bitácora de todas las modificaciones significativas realizadas en un proyecto.

3. Gestion de Tareas

- **Creación y Edición de Tareas:** Proceso de añadir nuevas tareas a un proyecto y modificar las existentes.
- **Asignación Manual de Tareas:** Proceso de asignar tareas a colaboradores de forma directa por un usuario.
- **Clasificación por Prioridad y Dificultad:** Proceso de establecer niveles de importancia y complejidad para las tareas.
- **Gestión de Estados de Tareas:** Proceso de cambio y visualización del progreso de las tareas (Pendiente, En progreso, Completada).
- **Movimiento de Tareas entre Columnas (Drag and Drop):** Proceso de cambio visual del estado de una tarea en un tablero mediante arrastrar y soltar.
- **Establecimiento de Fechas Límite:** Proceso de definir plazos para la finalización de las tareas.
- **Establecimiento de Dependencias entre Tareas:** Proceso de vincular tareas, indicando que una depende de la finalización de otra.
- **Gestión de Comentarios y Notas en Tareas:** Proceso de añadir y gestionar comunicaciones internas relacionadas con las tareas.

4. Procesamiento de Inteligencia Artificial

- **Análisis de Habilidades de Usuarios:** Proceso de recopilación y mejora del perfil de habilidades de los usuarios.
- **Algoritmo de Matching Tareas-Usuarios:** Proceso de emparejar tareas con los usuarios más adecuados basado en diversos criterios.
- **Interacción mediante Chat Interactivo con IA:** Proceso de comunicación con un chatbot para obtener información o realizar acciones.
- **Procesamiento de Lenguaje Natural para Descripción de Tareas:** Proceso de extracción de información clave de descripciones de tareas en texto libre.
- **Aprendizaje Automático Basado en Desempeño:** Proceso de mejora de las asignaciones y recomendaciones de IA a partir de resultados pasados.
- **Análisis Predictivo de Carga de Trabajo:** Proceso de proyección de la carga de trabajo futura de usuarios o equipos.
- **Optimización Automática de Distribución:** Proceso de sugerencia o aplicación de rebalanceo de tareas para optimizar la carga de trabajo.

5. Gestion de Colaboración

- **Sistema de Notificaciones:** Proceso de envío de alertas sobre eventos relevantes a los usuarios.
- **Chat y Mensajería entre Colaboradores:** Proceso de comunicación instantánea entre los miembros del equipo.
- **Comentarios y Feedback en Tareas:** Proceso de adición de comentarios y reacciones a las tareas.
- **Historial de Actividades del Proyecto:** Proceso de visualización de una línea de tiempo de todas las actividades realizadas en un proyecto.
- **Sistema de Menciones:** Proceso de etiquetado de otros colaboradores en comentarios o descripciones.
- **Compartir Archivos y Recursos:** Proceso de adjuntar y acceder a documentos y materiales relacionados con tareas o proyectos.
- **Sincronización en Tiempo Real:** Proceso de actualización inmediata de los datos para todos los usuarios.
- **Resolución de Conflictos de Edición:** Proceso de manejo de ediciones simultáneas para evitar la pérdida de datos.

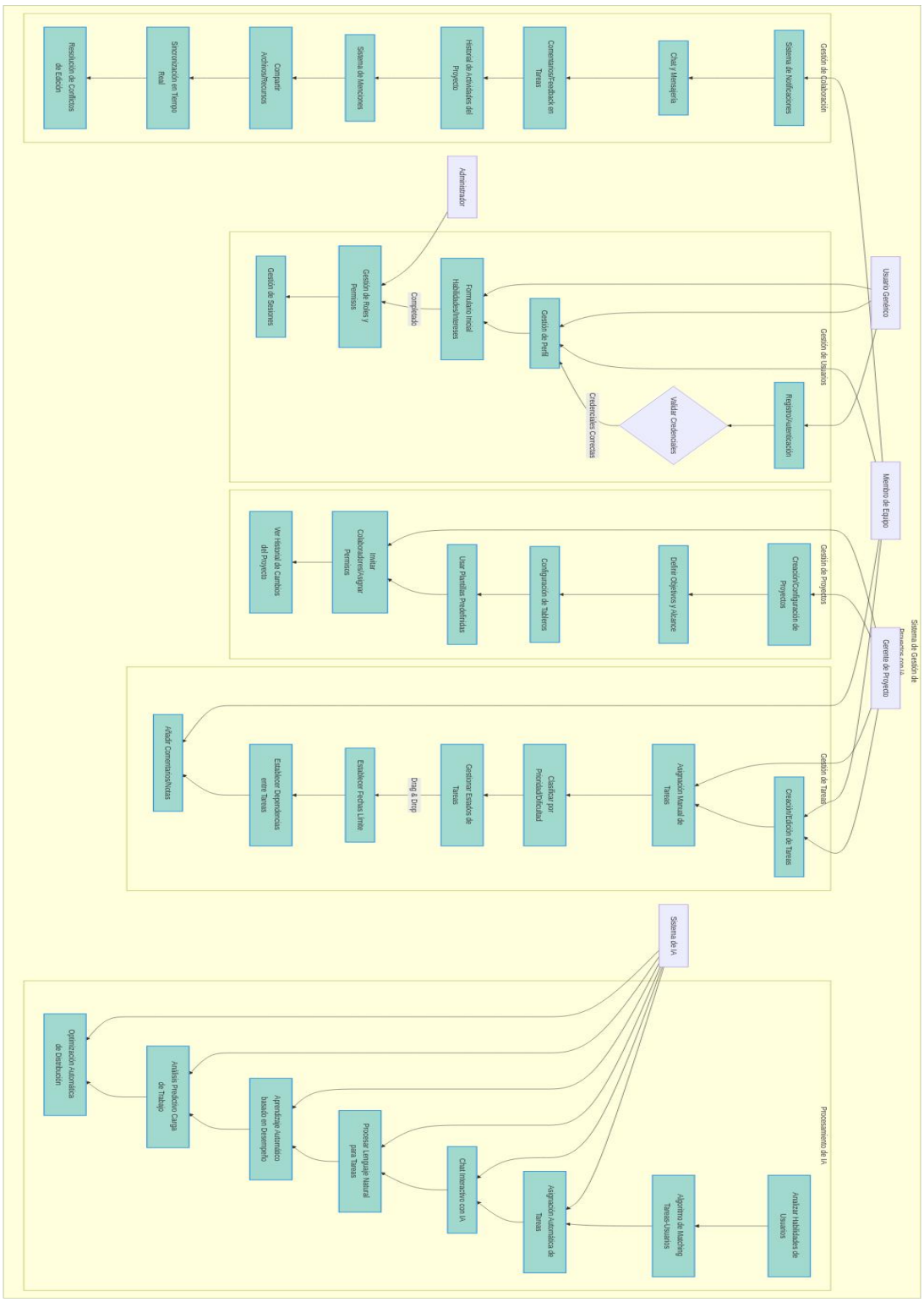
Determinación de Procesos Apoyados por la Aplicación.

Todos los procesos de negocio identificados serán apoyados directamente por la aplicación, ya que esta herramienta está diseñada para automatizar y mejorar la planificación, gestión y colaboración en proyectos, integrando capacidades de inteligencia artificial. La aplicación web centralizará y facilitará cada una de estas operaciones.

Diagrama de Procesos UML

Aquí se presenta un diagrama de actividades UML simplificado que representa los flujos principales del sistema, marcando con color los procesos clave que son asistidos o mejorados por la aplicación. Dada la complejidad de todos los procesos, se ha optado por un diagrama que visualiza la interacción de las principales funcionalidades con los usuarios y la IA.

Diagrama UML



Matriz de Actores vs Procesos

Actor funcional	Procesos que puede realizar directamente (extraídos del documento)
Usuario Regular *(estudiante, colaborador)*	Registro y autenticación de usuario · Edición del perfil · Llenar formulario inicial · Gestión de sesión · Visualizar tareas asignadas · Comentar tareas · Mover tareas en tableros · Adjuntar archivos · Mencionar compañeros · Consultar historial de actividades
Líder de Proyecto *(rol dentro de cada proyecto)*	Crear y configurar proyectos · Definir objetivos SMART · Usar plantillas predefinidas · Configurar tableros (Kanban, Scrum) · Gestionar colaboradores del proyecto · Establecer fechas límite · Clasificar tareas por prioridad y dificultad · Establecer dependencias · Supervisar cambios · Comentar tareas
Colaborador *(invitado a un proyecto)*	Visualizar tareas asignadas · Cambiar estados de tareas · Comentar y reaccionar · Adjuntar archivos · Usar menciones · Recibir notificaciones · Ver historial del proyecto
Administrador del Sistema	Crear/modificar/eliminar roles y permisos globales · Gestionar usuarios · Validar credenciales · Supervisar funcionamiento general del sistema · Resolver conflictos de edición · Controlar sesiones · Garantizar sincronización en tiempo real
Visitante *(sin cuenta registrada)*	Explorar contenido público · Consultar noticias o reglamento institucional *(si se habilita módulo informativo)*

Requisitos funcionales

1. Prioridad Alta

a) Requisitos Base del Sistema

- **RF 001 Registro y autenticación de usuarios:** Fundamental para que los usuarios puedan acceder al sistema.
- **RF 002 Gestión de perfiles:** Permite a los usuarios personalizar su información y es vital para la asignación de tareas.
- **RF 003 Roles y permisos:** Establece el control de acceso y las capacidades de cada usuario dentro de la aplicación.
- **RF 005 Sistema de Validación de credenciales:** Protege el acceso y la seguridad de las cuentas de usuario.
- **RF 006 Gestión de sesiones:** Asegura que la sesión de los usuarios sea segura y controlada, evitando accesos no autorizados.

b) Gestion de Proyectos

- **RF 007 Creación y configuración de proyectos:** Permite a los usuarios crear y establecer la estructura de su trabajo.
- **RF 011 Colaboradores y permisos por proyecto:** Permite la colaboración en proyectos y define quién puede hacer qué.

c) Gestion de Tareas

- **RF 013 Creación y edición de tareas:** La funcionalidad central del sistema, permitiendo a los usuarios gestionar su trabajo.
- **RF 014 Asignación manual de tareas:** Permite la distribución de trabajo entre los colaboradores de un proyecto.

- **RF 017 Estados de tareas:** Permite el seguimiento del progreso de las tareas, esencial para la planificación y visualización.
- **RF 019 Fechas límite:** Permite establecer plazos, lo que es crucial para la gestión de proyectos y la organización.

d) Gestion de Colaboracion

- **RF 029 Sistema de notificaciones:** Mantiene a los usuarios informados sobre eventos importantes en los proyectos.
- **RF 031 Comentarios y Feedback en tareas:** Facilita la comunicación y el seguimiento del trabajo en tareas específicas.
- **RF 035 Sincronización en tiempo real:** Garantiza que todos los colaboradores vean la información actualizada al instante, mejorando la colaboración.

2. Prioridad Media

a) Gestion de Usuarios

- **RF 004 Formulario inicial:** Recopila información clave después del registro, mejorando la personalización del perfil.

b) Gestion de Proyectos

- **RF 008 Definición de objetivos y alcance:** Permite estructurar proyectos de manera más formal y detallada.
- **RF 009 Configuración de tableros:** Ofrece flexibilidad en la visualización de las tareas (ej. Kanban).
- **RF 012 Historial de cambios del proyecto:** Proporciona un registro de las actividades, útil para auditorías y seguimiento.

c) Gestion de Tareas

- **RF 015 Asignación automática mediante IA:** Sugiere la asignación de tareas, una mejora significativa pero no básica.
- **RF 016 Clasificación por prioridad y dificultad:** Permite organizar y filtrar las tareas según su importancia y complejidad.
- **RF 018 Drag and Drop entre columnas:** Mejora la usabilidad al permitir mover tareas entre estados de forma intuitiva.
- **RF 020 Dependencia entre tareas:** Permite gestionar la secuencia de trabajo en proyectos complejos.
- **RF 021 Comentarios y notas entre tareas:** Permite adjuntar notas y comentarios, mejorando la comunicación.

d) Gestion de colaboracion

- **RF 030 Chat y mensajería entre colaboradores:** Habilita la comunicación directa en la plataforma.
- **RF 032 Historial de actividades del proyecto:** Proporciona una línea de tiempo de todas las acciones del proyecto.
- **RF 033 Sistema de menciones:** Facilita la comunicación y la atención a usuarios específicos en comentarios.
- **RF 034 Compartir archivos y recursos:** Permite la gestión de documentos directamente en el proyecto.
- **RF 036 Resolución de conflictos de edición:** Evita la pérdida de datos cuando múltiples usuarios editan el mismo elemento.

3. Prioridad Baja

a) Gestion de Proyectos

- **RF 010 Plantillas de proyectos predefinidas:** Facilita la creación de proyectos al ofrecer estructuras pre-cargadas

b) Procesamiento de IA

- **RF 022 Análisis de habilidades de usuarios:** Recopila información para el algoritmo de matching.
- **RF 023 Algoritmo de matching tareas-usuarios:** La lógica para sugerir asignaciones de tareas basadas en habilidades y carga de trabajo.
- **RF 024 Chat interactivo con IA:** Un chatbot que puede responder preguntas y realizar acciones simples.
- **RF 025 Procesamiento de lenguaje natural para descripción de tareas:** Permite que la IA interprete las descripciones de las tareas para sugerir campos.
- **RF 026 Aprendizaje automático basado en desempeño:** Permite a la IA mejorar sus recomendaciones con el tiempo.
- **RF 027 Análisis predictivo de carga de trabajo:** Permite predecir la carga de trabajo futura para evitar sobrecargas.
- **RF 028 Optimización automática de distribución:** Sugiere o aplica reequilibrios en la carga de trabajo de los equipos.

Requisitos No Funcionales

1. Prioridad Alta

a) Rendimiento

- RNF REND 001 (registro en menos de 30s)
- RNF REND 003 (validación en menos de 30s)
- RNF REND 011 (creación/edición de tareas en menos de 10s)
- RNF REND 018 (sincronización en tiempo real)

b) Seguridad

- RNF SEG 001 (hashing de contraseñas, HTTPS)
- RNF SEG 002 (control de acceso basado en roles)
- RNF SEG003 (validación robusta de credenciales)
- RNF SEG 004 (seguridad de sesiones).

c) Usabilidad

- RNF USAB 001 (mensajes de error claros)
- RNF USAB 002 (interfaz de perfil intuitiva)
- RNF USAB 008 (interfaz de tareas intuitiva)

d) Fiabilidad

- RNF FIAB 001 (disponibilidad del 99.9%)
- RNF FIAB 002 (consistencia de roles y permisos)
- RNF FIAB 012 (sincronización de datos precisa)
- RNF FIAB 013 (prevención de pérdida de datos por ediciones concurrentes).

2. Prioridad Media

a) Rendimiento

- RNF REND 005 (creación de proyectos en menos de 20s)
- RNF REND 009 (registro de cambios en menos de 20s)

b) Usabilidad

- RNF USAB 004 (interfaz de proyectos intuitiva)
- RNF USAB 012 (funcionalidad de Drag and Drop fluida)
- RNF USAB014(interfaz de comentarios clara)

c) Mantenibilidad

- RNF MANT 001 (código documentado)
- RNF MANT 002 (gestión de roles fácil de modificar)

d) Compatibilidad

- RNF COMP 001 (accesible desde diferentes navegadores y dispositivos).

3. Prioridad Baja

a) Rendimientos

- RNF REND 015 (algoritmo de matching en menos de 30s).

b) Seguridad

- RNF SEG 006 (integridad de asignaciones automáticas).

c) Usabilidad

- RNF USAB 016 (interfaz de chatbot intuitiva).

d) Fiabilidad

- RNF FIAB 011 (mejoras estadísticas en recomendaciones de IA).

e) Mantenibilidad

- RNF MANT 006 (actualizaciones de IA sin interrupciones).

f) Compatibilidad

- RNF COMP 002 (formulario compatible con móviles y escritorio).

Requisitos Funcionales y No funcionales Volere

Requisitos Funcionales Clave		Requisitos No Funcionales Clave	
Gestión de Usuarios	El sistema debe permitir el registro, la autenticación y la gestión de perfiles de los usuarios.También es crucial la asignación de roles y permisos, junto con la validación de credenciales y la gestión de sesiones.	Rendimiento	El sistema debe procesar acciones como el registro, la autenticación, la validación de credenciales y la creación de tareas en un tiempo razonable para una experiencia de usuario fluida.La sincronización en tiempo real no debe tener latencia perceptible.
Gestión de Proyectos	El proyecto debe poder ser creado y configurado.Es	Seguridad	Es indispensable implementar el hashing de

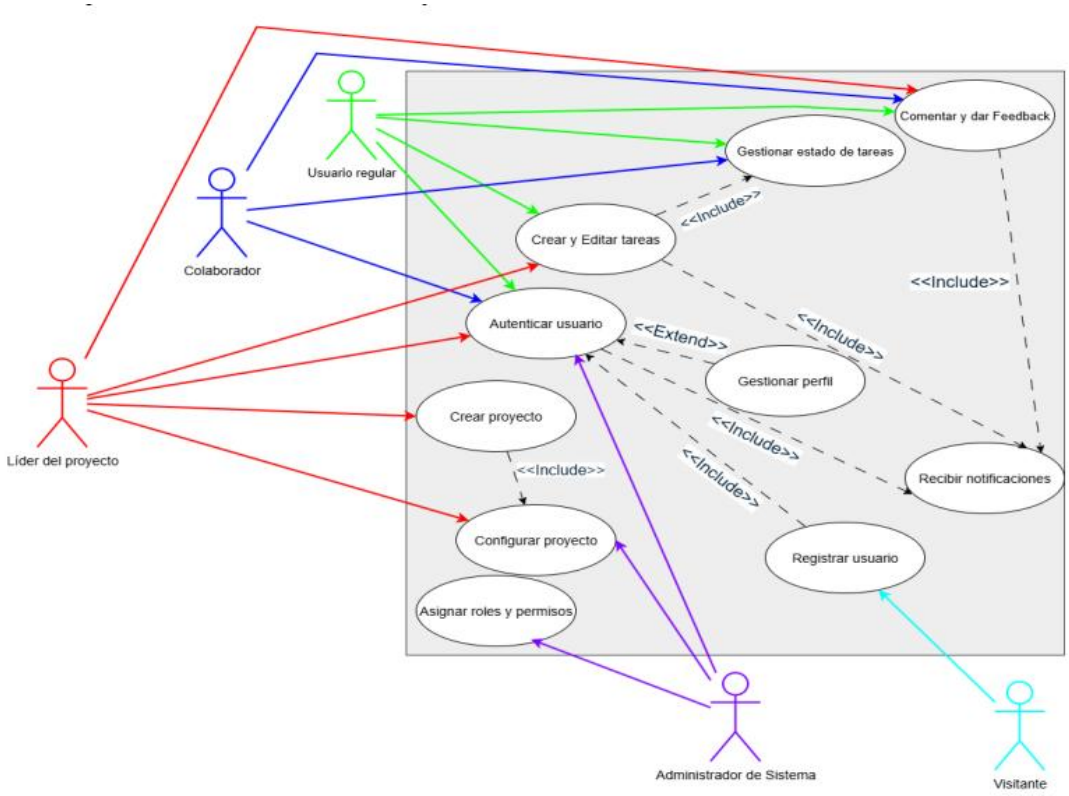
	fundamental poder añadir colaboradores a un proyecto y asignarles permisos específicos.		contraseñas y el uso de HTTPS.Debe haber un control de acceso basado en roles para restringir las acciones de los usuarios.Las sesiones de usuario deben estar protegidas.
Gestión de Tareas	La funcionalidad central es la creación y edición de tareas.Se debe poder asignar tareas de forma manual ,y el sistema debe gestionar los estados delas tareas.También se deben poder establecer fechas límite para las tareas.	Usabilidad	La interfaz debe ser intuitiva y fácil de usar, con mensajes de error claros.Funcionalidades clave como la gestión de perfiles, la creación de proyectos y la gestión de tareas deben ser sencillas y sin complicaciones.

Gestión de Colaboración	El sistema debe tener un sistema de notificaciones para eventos relevantes y permitir a los colaboradores dejar comentarios y Feedback en las tareas. La sincronización en tiempo real es fundamental para una colaboración efectiva.	Fiabilidad	El sistema debe estar disponible la mayor parte del tiempo (99.9%) y garantizar que las asignaciones de roles y permisos se apliquen correctamente. La sincronización de datos debe ser precisa para evitar la pérdida de información.

Contexto del Escenario

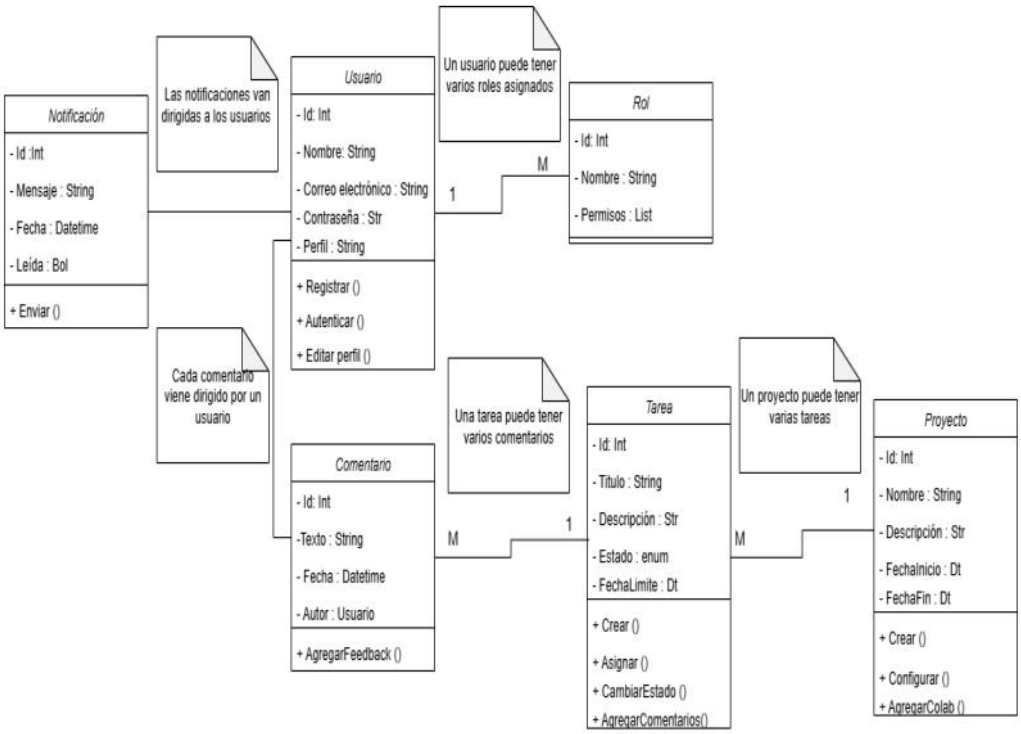
En una universidad reconocida por su enfoque en innovación tecnológica como lo es la UNETI, un grupo diverso compuesto por estudiantes se reúne para abordar un proyecto complejo que requiere múltiples disciplinas y habilidades técnicas. Para ello deciden utilizar la aplicación que has desarrollado, la cual está diseñada específicamente para optimizar la gestión y ejecución de proyectos y actividades apoyados por inteligencia artificial para optimizar la subdivisión de tareas y la comunicación.

Diagrama de Casos de Uso



Para facilitar el seguimiento se coloreó a cada actor involucrado.

Diagrama de Clases



Estrategias para Validar Requisitos

1. Revisiones de Requisitos:

a) Revisiones por Pares:

- Involucrar a otros miembros del equipo para revisar los requisitos. Esto ayuda a identificar inconsistencias y omisiones.
- Realizar reuniones periódicas donde se discutan los requisitos y se obtenga retro alimentación.

b) Revisiones con Partes Interesadas / Interesados Clave:

- Presentar los requisitos a los interesados para asegurarse de quereflejan sus necesidades.
- Utilizar un formato claro y conciso para facilitar la comprensión.

2. Entrevistas:

a) Entrevistas a Usuarios Finales:

- Realizar entrevistas con los usuarios finales para comprender sus necesidades y expectativas.
- Formular preguntas abiertas que permitan a los usuarios expresar sus opiniones y sugerencias.

b) Entrevistas con Expertos:

- Consultar a expertos en el área para obtener su perspectiva sobre la viabilidad y relevancia de los requisitos.

3. Prototipos:

a) Prototipos de Baja Fidelidad:

- Crear bocetos o wireframes que representen la interfaz y funcionalidad del sistema. Esto permite a los usuarios visualizar el producto y proporcionar retroalimentación temprana.

b) Prototipos de Alta Fidelidad:

- Desarrollar versiones más avanzadas del prototipo que incluyan interactividad. Esto ayuda a los usuarios a experimentar el sistema de manera más realista y a identificar problemas potenciales.

4. Pruebas de Usabilidad:

a) Sesiones de Pruebas con Usuarios:

- Realizar pruebas de usabilidad donde los usuarios interactúan con el prototipo y se observa su comportamiento.
- Recoger datos sobre la facilidad de uso y la satisfacción del usuario.

5. Documentación y Seguimientos:

a) Registro de Cambios:

- Mantener un registro de todos los cambios realizados en los requisitos a lo largo del proyecto. Esto ayuda a rastrear la evolución de los requisitos y a justificar decisiones.

b) Revisiones Continuas:

- Establecer un proceso de revisión continua a lo largo del ciclo de vida del proyecto para asegurar que los requisitos sigan siendo relevantes y se ajusten a las necesidades cambiantes.

Definición de Seguimientos y Control de Cambios

Git es una herramienta poderosa para el control de versiones que permite gestionar cambios en el código y la documentación de un proyecto de manera eficiente. A continuación, se describen las estrategias para implementar el seguimiento y control de cambios utilizando Git.

1. Establecimiento de un Repositorio Git

a) Creación del Repositorio:

- Iniciar un repositorio Git para el proyecto, ya sea en una plataforma como GitHub, GitLab o Bitbucket.
- Asegurarse de que todos los miembros del equipo tengan acceso al repositorio.

b) Estructura del Repositorio:

- Organizar el repositorio con una estructura de carpetas clara que separe el código, la documentación y otros recursos del proyecto.
- Incluir un archivo README que explique el propósito del proyecto y cómo contribuir.

2. Control de Cambios

a) Commits Frecuentes:

- Realizar commits frecuentes con mensajes claros y descriptivos que expliquen el propósito de cada cambio.
- Seguir una convención de mensajes de commit para facilitar la comprensión del historial de cambios.

b) Ramas (Branches)

- Utilizar ramas para desarrollar nuevas características o realizar cambios significativos sin afectar la rama principal.
- Nombrar las ramas de manera descriptiva.

3. Revisión de Cambios

a) Pull Request:

- Implementar un proceso de Pull Requests para que los cambios en las ramas sean revisados por otros miembros del equipo antes de ser fusionados a la rama principal.
- Utilizar comentarios en los PR para discutir cambios y sugerir mejoras.

b) Revisiones de Código:

- Fomentar revisiones de código como parte del proceso de PR, donde los miembros del equipo pueden proporcionar retroalimentación sobre el código y los requisitos implementados.

4. Documentación de Cambios

a) Changelog:

- Mantener un archivo CHANGELOG.md en el repositorio que documente los cambios significativos en cada versión del proyecto.
- Incluir detalles sobre nuevas características, correcciones de errores y cambios en los requisitos.

b) Etiquetas:

- Utilizar etiquetas para marcar versiones específicas del proyecto, facilitando el seguimiento de cambios a lo largo del tiempo.

- Seguir un esquema de versionado semántico, por ejemplo v1.0.0, para las etiquetas.

5. Seguimiento del Progreso

a) Historial de Commits:

- Utilizar el historial de commits de Git para rastrear cambios y entender la evolución del proyecto.
- Ejecutar comandos como git log para ver el historial de cambios y git diff para comparar diferencias entre versiones.

Identificación de Riesgos y Estrategias de Mitigación

1. Riesgo Tecnicos

a) Riesgo: Fallos en la Tecnología

- **Descripción:** Dependencia de tecnologías que pueden fallar o no cumplir con las expectativas.
- **Mitigación:**
  - Realizar pruebas exhaustivas de las tecnologías seleccionadas antes de su implementación.
  - Mantener una lista de tecnologías alternativas que puedan ser utilizadas en caso de fallos.

b) Riesgo: Integración de Sistemas

- **Descripción:** Dificultades al integrar diferentes sistemas o componentes del proyecto
- **Mitigación:**
  - Planificar y realizar pruebas de integración desde las primeras etapas del desarrollo.
  - Documentar claramente las interfaces y protocolos de comunicación entre sistemas.

2. Riesgos de Requisitos

a) Riesgo: Cambios en los Requisitos:

- **Descripción:** Cambios inesperados en los requisitos que pueden afectar el alcance y el cronograma del proyecto.
- **Mitigación:**

- Establecer un proceso formal de control de cambios para gestionar las solicitudes de modificación de requisitos.
- Involucrar a los interesados en revisiones periódicas para asegurar que los requisitos se mantengan alineados con sus necesidades.

b) Riesgo: Requisitos Incompletos o Mal Entendidos:

- **Descripción:** Requisitos que no están bien definidos o que son malinterpretados por el equipo.
- **Mitigación:**
  - Realizar sesiones de validación de requisitos con los interesados para aclarar y confirmar los requisitos
  - Utilizar prototipos y revisiones para asegurar que los requisitos sean comprendidos correctamente.

3. Riesgos de Gestión del Proyecto

a) Riesgo: Retrasos en el Cronograma:

- **Descripción:** El proyecto puede no cumplir con los plazos establecidos debido a diversos factores.
- **Mitigación:**
  - Establecer un cronograma realista con márgenes de tiempo para imprevistos.
  - Realizar revisiones periódicas del progreso y ajustar el cronograma según sea necesario.

b) Riesgo: Falta de Recursos:

- **Descripción:** Insuficiencia de recursos humanos, técnicos o financieros para completar el proyecto.

- **Mitigación:**
  - Planificar y asignar recursos de manera efectiva desde el inicio del proyecto.
  - Identificar y asegurar recursos adicionales que puedan ser necesarios a lo largo del proyecto.

4. Riesgos de Comunicación

a) Riesgo: Falta de Comunicación entre el Equipo:

- **Descripción:** La falta de comunicación puede llevar a malentendidos y errores en el desarrollo.
- **Mitigacion:**
  - Establecer canales de comunicación claros y regulares, como reuniones diarias o semanales.
  - Utilizar herramientas de gestión de proyectos para mantener a todos los miembros del equipo informados sobre el progreso y los cambios.

b) Riesgo: Desalineación con los Interesados:

- **Descripción:** Los interesados pueden no estar alineados con el progreso o los resultados del proyecto.
- **Mitigación:**
  - Realizar reuniones periódicas de seguimiento con los interesados para mantenerlos informados y obtener su retroalimentación.
  - Documentar y comunicar claramente los objetivos y el progreso del proyecto.

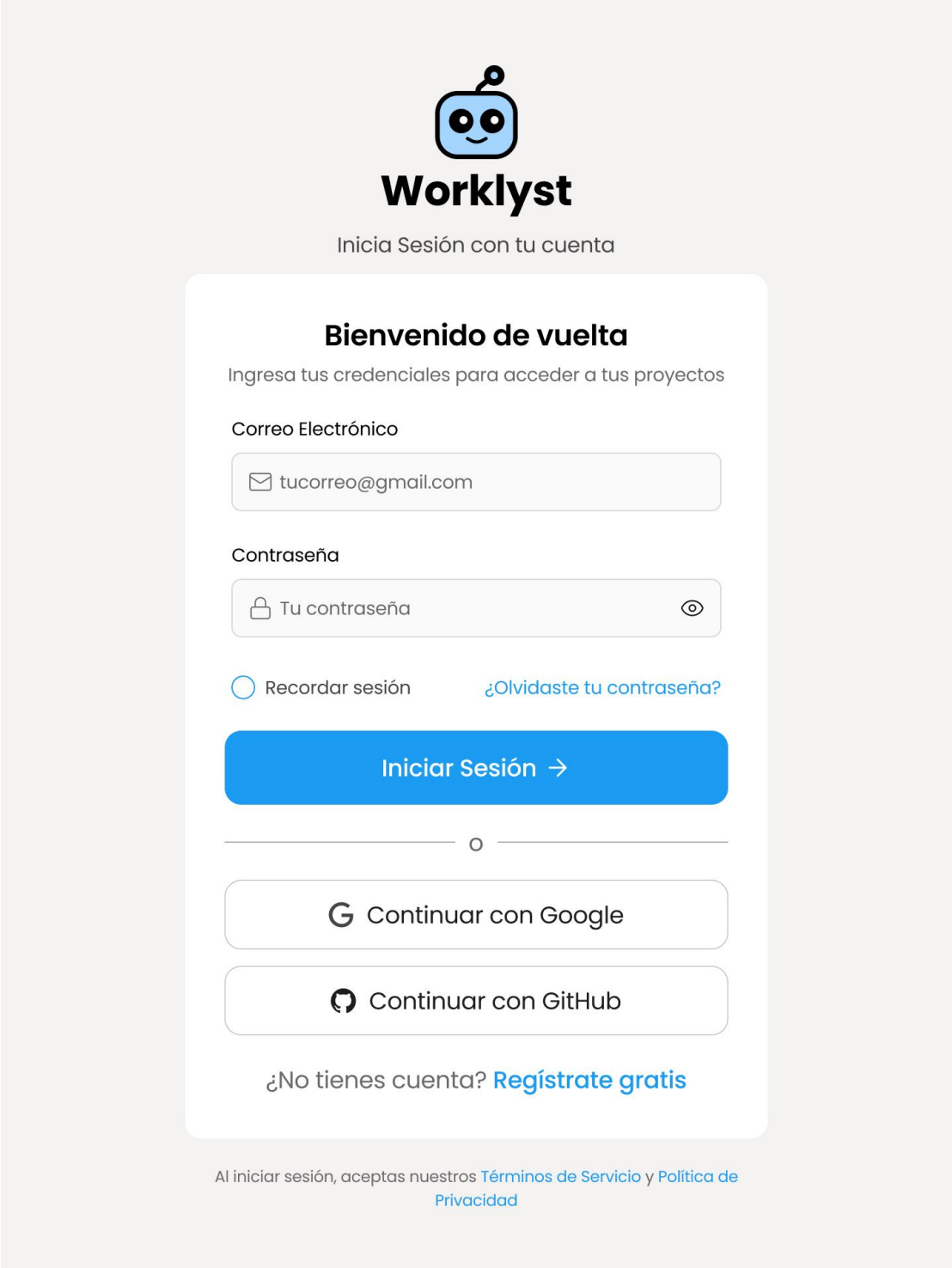
5. Riesgo de Calidad

a) Riesgo: Baja Calidad del Producto Final:

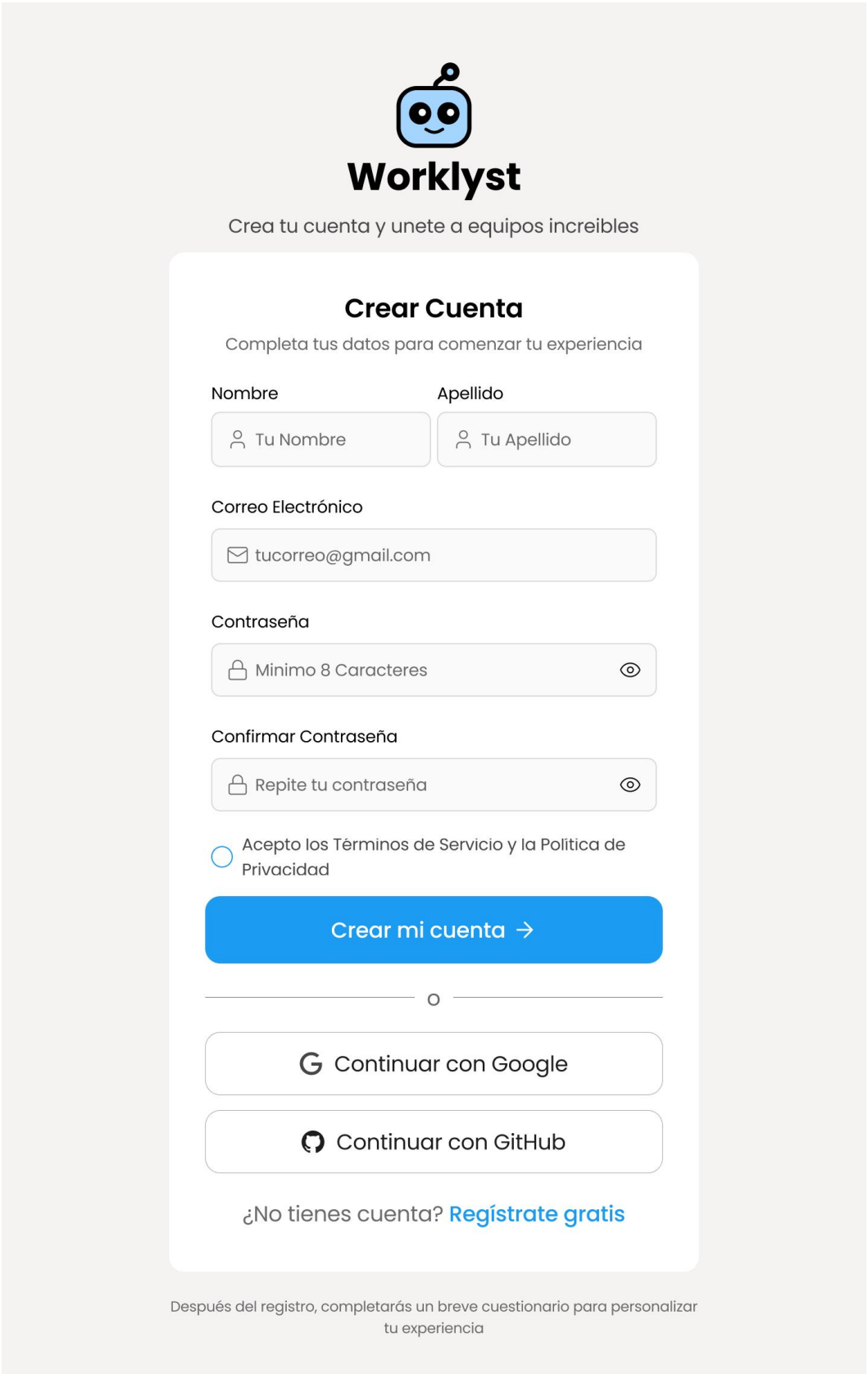
- **Descripción:** El producto final puede no cumplir con los estándares de calidad esperados.
- **Mitigacion:**
  - Implementar un proceso de control de calidad que incluya pruebas regulares y revisiones del producto.
  - Fomentar una cultura de calidad dentro del equipo, donde todos sean responsables de la calidad del trabajo.

Interfaz Gráfica de la Aplicación

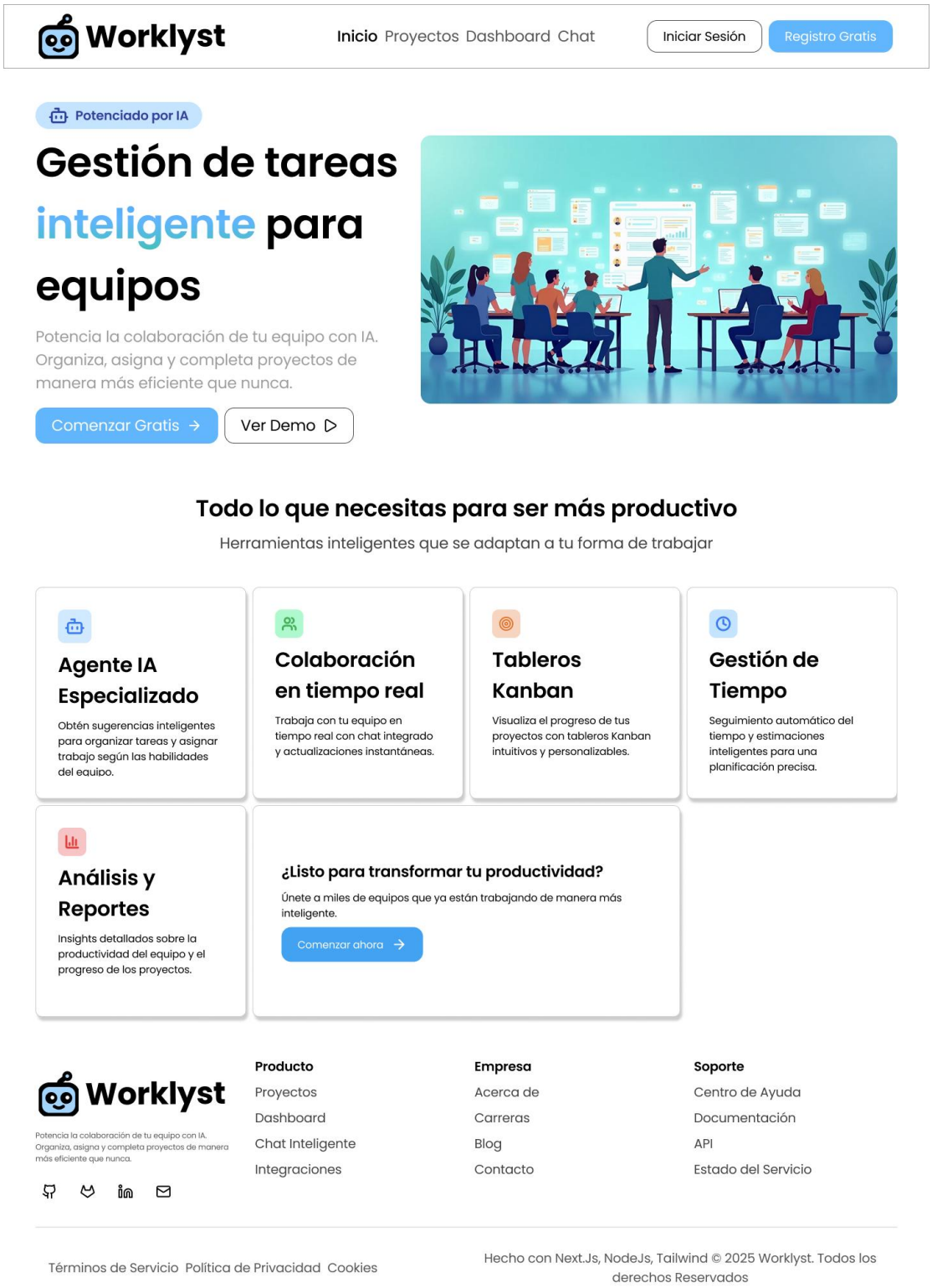
Login



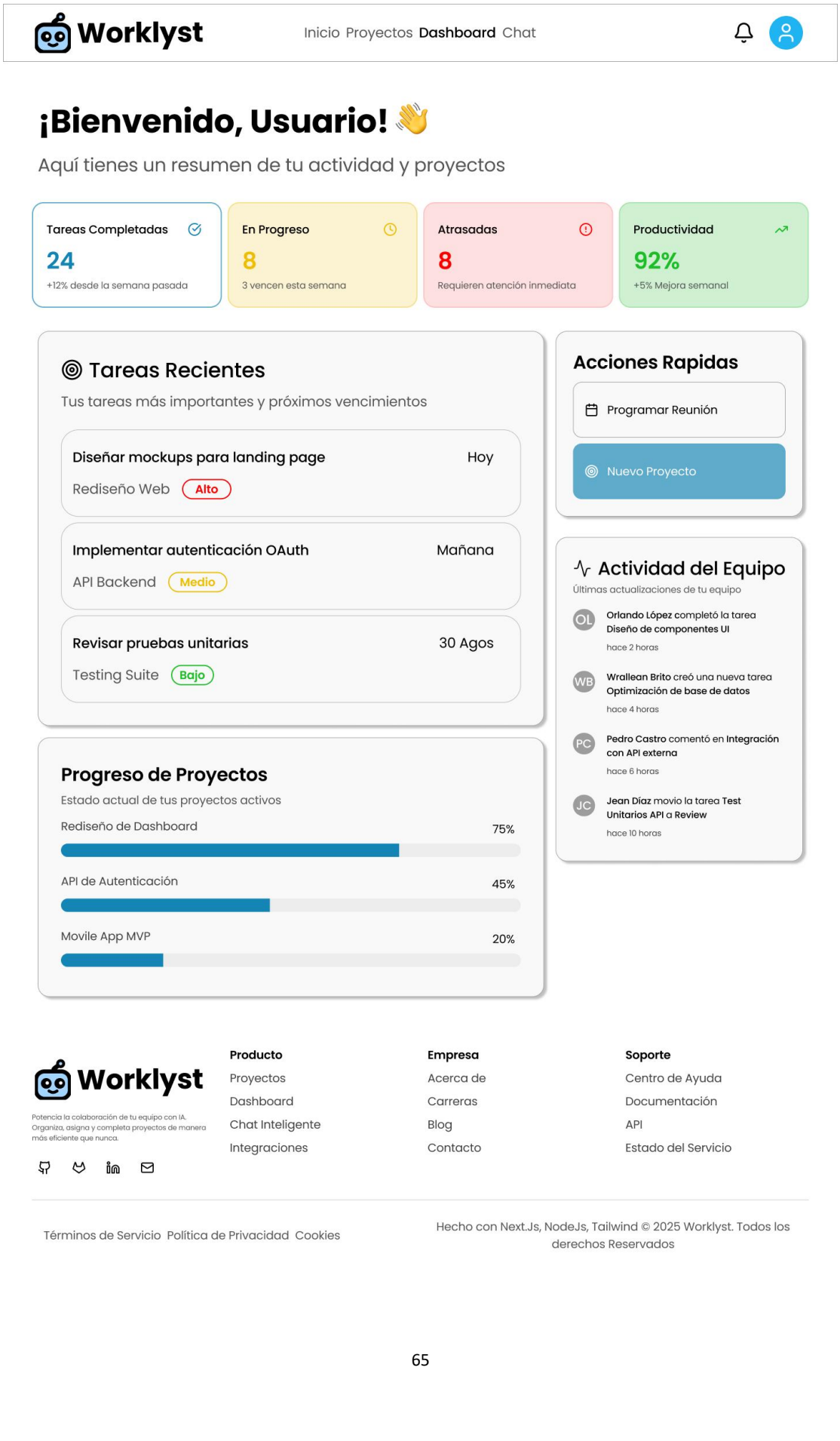
Registro


Worklyst
Crea tu cuenta y unete a equipos increibles
Crear Cuenta
Completa tus datos para comenzar tu experiencia
Nombre
Tu Nombre
Apellido
Tu Apellido
Correo Electrónico
tucorreo@gmail.com
Contraseña
Minimo 8 Caracteres
Confirmar Contraseña
Repite tu contraseña
Acepto los Términos de Servicio y la Política de Privacidad
Crear mi cuenta →
O
Continuar con Google
Continuar con GitHub
¿No tienes cuenta? Regístrate gratis
Después del registro, completarás un breve cuestionario para personalizar tu experiencia

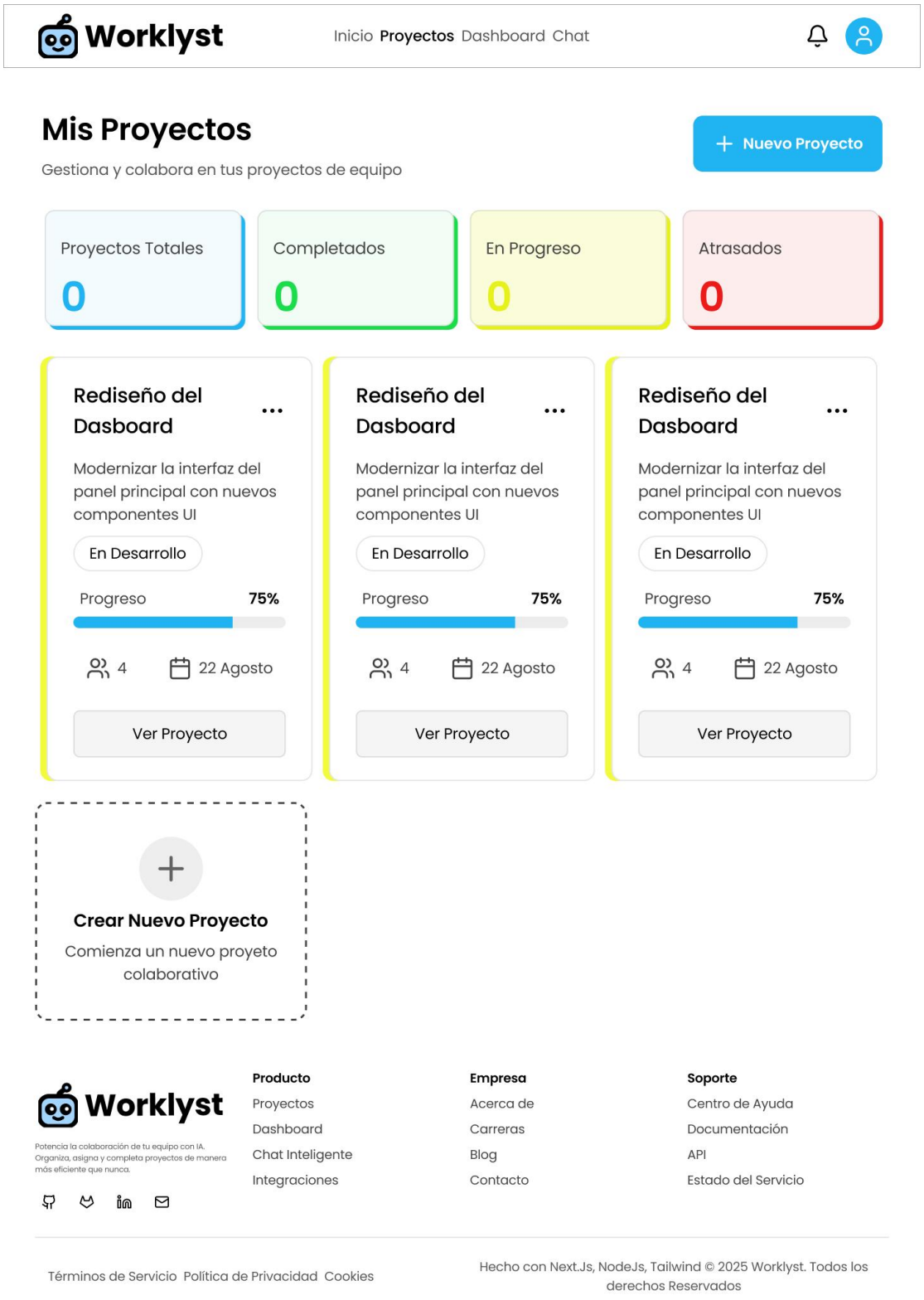
Home


Worklyst
Inicio Proyectos Dashboard Chat
Iniciar Sesión
Registro Gratis
Potenciado por IA
Gestión de tareas inteligente para equipos
Potencia la colaboración de tu equipo con IA. Organiza, asigna y completa proyectos de manera más eficiente que nunca.
Comenzar Gratis
Ver Demo
Todo lo que necesitas para ser más productivo
Herramientas inteligentes que se adaptan a tu forma de trabajar
Agente IA Especializado
Obtén sugerencias inteligentes para organizar tareas y asignar trabajo según las habilidades del equipo.
Colaboración en tiempo real
Trabaja con tu equipo en tiempo real con chat integrado y actualizaciones instantáneas.
Tableros Kanban
Visualiza el progreso de tus proyectos con tableros Kanban intuitivos y personalizables.
Gestión de Tiempo
Seguimiento automático del tiempo y estimaciones inteligentes para una planificación precisa.
Análisis y Reportes
Insights detallados sobre la productividad del equipo y el progreso de los proyectos.
¿Listo para transformar tu productividad?
Únete a miles de equipos que ya están trabajando de manera más inteligente.
Comenzar ahora
Worklyst
Potencia la colaboración de tu equipo con IA. Organiza, asigna y completa proyectos de manera más eficiente que nunca.
Producto
Proyectos
Dashboard
Chat Inteligente
Integraciones
Empresa
Acerca de
Carreras
Blog
Contacto
Soporte
Centro de Ayuda
Documentación
API
Estado del Servicio
Términos de Servicio Política de Privacidad Cookies
Hecho con Next.Js, NodeJs, Tailwind © 2025 Worklyst. Todos los derechos Reservados

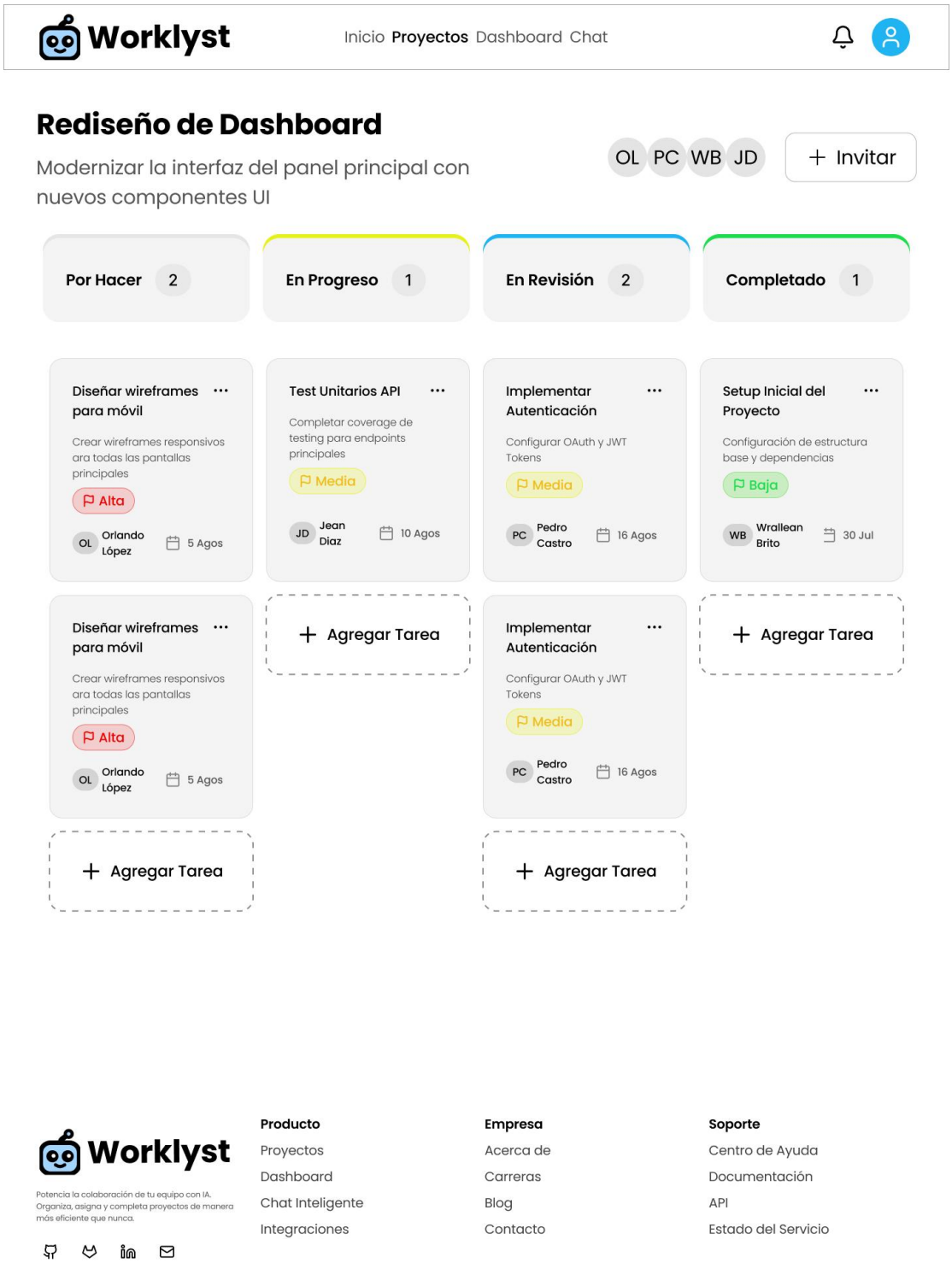
Dashboard


Worklyst
Inicio Proyectos Dashboard Chat
¡Bienvenido, Usuario! 👋
Aquí tienes un resumen de tu actividad y proyectos
Tareas Completadas
24
+12% desde la semana pasada
En Progreso
8
3 vencen esta semana
Atrasadas
8
Requieren atención inmediata
Productividad
92%
+5% Mejora semanal
Tareas Recientes
Tus tareas más importantes y próximos vencimientos
Diseñar mockups para landing page
Hoy
Rediseño Web
Alto
Implementar autenticación OAuth
Mañana
API Backend
Medio
Revisar pruebas unitarias
30 Agos
Testing Suite
Bajo
Acciones Rapidas
Programar Reunión
Nuevo Proyecto
Actividad del Equipo
Últimas actualizaciones de tu equipo
OL
Orlando López completó la tarea Diseño de componentes UI
hace 2 horas
WB
Wrallean Brito creó una nueva tarea Optimización de base de datos
hace 4 horas
PC
Pedro Castro comentó en Integración con API externa
hace 6 horas
JC
Jean Díaz movio la tarea Test Unitarios API a Review
hace 10 horas
Progreso de Proyectos
Estado actual de tus proyectos activos
Rediseño de Dashboard
75%
API de Autenticación
45%
Movile App MVP
20%
Worklyst
Potencia la colaboración de tu equipo con IA. Organiza, asigna y completa proyectos de manera más eficiente que nunca.
Producto
Proyectos
Dashboard
Chat Inteligente
Integraciones
Empresa
Acerca de
Carreras
Blog
Contacto
Soporte
Centro de Ayuda
Documentación
API
Estado del Servicio
Términos de Servicio Política de Privacidad Cookies
Hecho con Next.Js, NodeJs, Tailwind © 2025 Worklyst. Todos los derechos Reservados

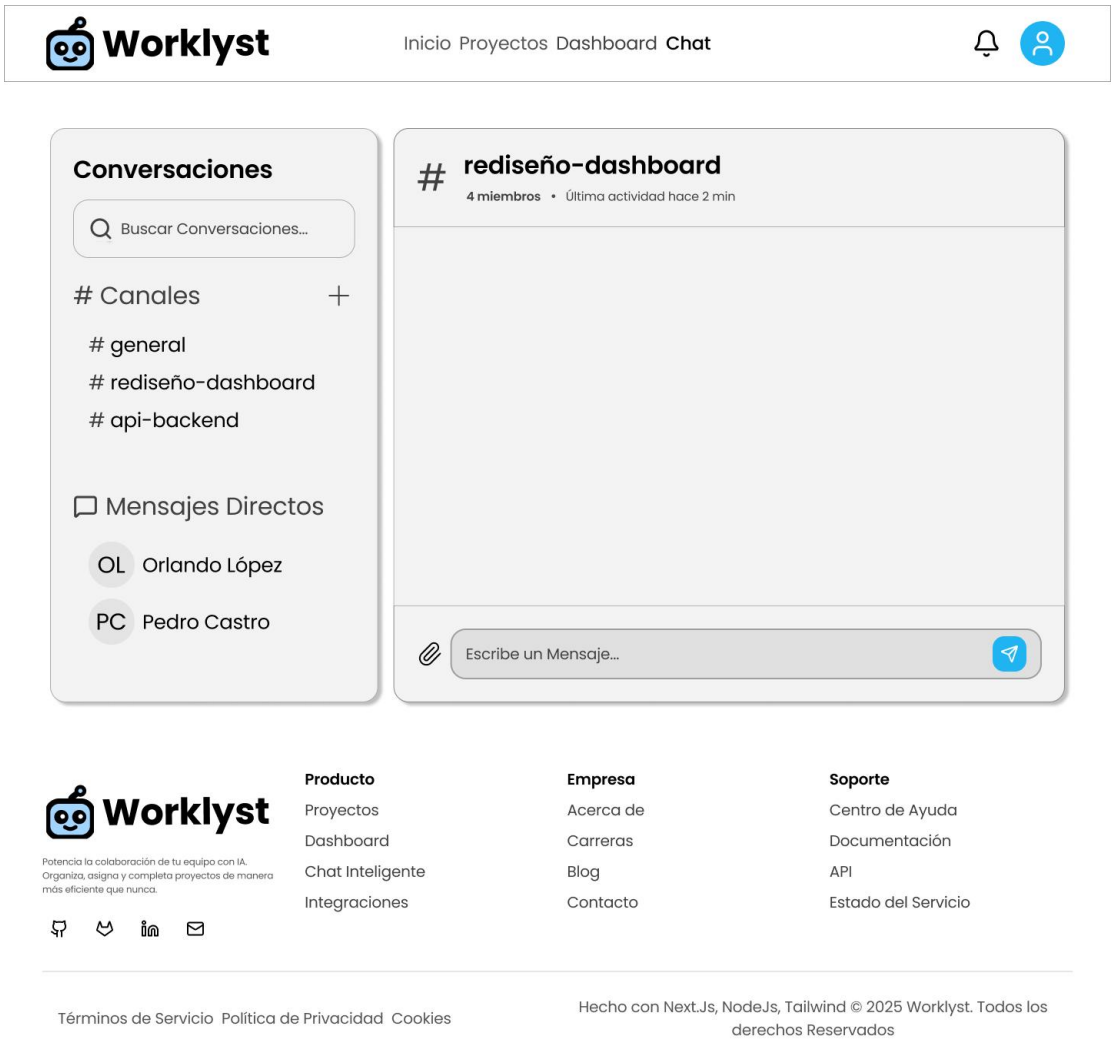
Proyectos


Worklyst
Inicio Proyectos Dashboard Chat
Mis Proyectos
Gestiona y colabora en tus proyectos de equipo
Nuevo Proyecto
Proyectos Totales
0
Completados
0
En Progreso
0
Atrasados
0
Rediseño del Dasboard
Modernizar la interfaz del panel principal con nuevos componentes UI
En Desarrollo
Progreso 75%
4
22 Agosto
Ver Proyecto
Rediseño del Dasboard
Modernizar la interfaz del panel principal con nuevos componentes UI
En Desarrollo
Progreso 75%
4
22 Agosto
Ver Proyecto
Rediseño del Dasboard
Modernizar la interfaz del panel principal con nuevos componentes UI
En Desarrollo
Progreso 75%
4
22 Agosto
Ver Proyecto
Crear Nuevo Proyecto
Comienza un nuevo proyeto colaborativo
Worklyst
Potencia la colaboración de tu equipo con IA. Organiza, asigna y completa proyectos de manera más eficiente que nunca.
Producto
Proyectos
Dashboard
Chat Inteligente
Integraciones
Empresa
Acerca de
Carreras
Blog
Contacto
Soporte
Centro de Ayuda
Documentación
API
Estado del Servicio
Términos de Servicio Política de Privacidad Cookies
Hecho con Next.Js, NodeJs, Tailwind © 2025 Worklyst. Todos los derechos Reservados

Proyectos


Worklyst
Inicio Proyectos Dashboard Chat
Rediseño de Dashboard
Modernizar la interfaz del panel principal con nuevos componentes UI
OL PC WB JD
+ Invitar
Por Hacer 2
En Progreso 1
En Revisión 2
Completado 1
Diseñar wireframes para móvil
Crear wireframes responsivos ara todas las pantallas principales
Alta
OL Orlando López
5 Agos
Test Unitarios API
Completar coverage de testing para endpoints principales
Media
JD Jean Diaz
10 Agos
Implementar Autenticación
Configurar OAuth y JWT Tokens
Media
PC Pedro Castro
16 Agos
Setup Inicial del Proyecto
Configuración de estructura base y dependencias
Baja
WB Wrallean Brito
30 Jul
+ Agregar Tarea
+ Agregar Tarea
+ Agregar Tarea
Worklyst
Potencia la colaboración de tu equipo con IA. Organiza, asigna y completa proyectos de manera más eficiente que nunca.
Producto
Proyectos
Dashboard
Chat Inteligente
Integraciones
Empresa
Acerca de
Carreras
Blog
Contacto
Soporte
Centro de Ayuda
Documentación
API
Estado del Servicio
Términos de Servicio Política de Privacidad Cookies
Hecho con Next.Js, NodeJs, Tailwind © 2025 Worklyst. Todos los derechos Reservados

ChatBot IA


Worklyst
Inicio Proyectos Dashboard Chat
Conversaciones
Buscar Conversaciones...
Canales
general
rediseño-dashboard
api-backend
Mensajes Directos
OL Orlando López
PC Pedro Castro
rediseño-dashboard
4 miembros • Última actividad hace 2 min
Escribe un Mensaje...
Worklyst
Potencia la colaboración de tu equipo con IA. Organiza, asigna y completa proyectos de manera más eficiente que nunca.
Producto
Proyectos
Dashboard
Chat Inteligente
Integraciones
Empresa
Acerca de
Carreras
Blog
Contacto
Soporte
Centro de Ayuda
Documentación
API
Estado del Servicio
Términos de Servicio Política de Privacidad Cookies
Hecho con Next.Js, NodeJs, Tailwind © 2025 Worklyst. Todos los derechos Reservados

Proposito del Proyecto

Worklyst es una plataforma web diseñada para facilitar la gestión colaborativa de proyectos mediante inteligencia artificial. Su objetivo principal es optimizar la organización de tareas, mejorar la asignación de responsabilidades y ofrecer una experiencia visual clara y adaptable para equipos multidisciplinarios.

Vision

Worklyst busca convertirse en un asistente inteligente para equipos creativos y técnicos, capaz de entender el contexto de cada proyecto, proponer acciones estratégicas y facilitar la toma de decisiones. La plataforma combina diseño emocional, modularidad técnica y razonamiento automatizado para ofrecer una experiencia fluida, intuitiva y escalable.

Enfoque basado en IA y colaboración

El núcleo de Worklyst está impulsado por un agente IA que interpreta comandos naturales, divide proyectos en tareas, asigna miembros según habilidades y mueve tareas entre columnas. Este enfoque permite que los equipos trabajen de forma más autónoma, con menos fricción y mayor claridad. La colaboración se potencia mediante interfaces compartidas, trazabilidad de decisiones y sugerencias inteligentes que respetan el estilo de trabajo de cada usuario.

Arquitectura del Software

La arquitectura de Worklyst está diseñada para ser modular, escalable y adaptable a equipos multidisciplinarios. Se compone de cuatro capas principales que se comunican entre sí de forma eficiente:

1. Componentes Principales

Frontend: desarrollado con Next.js, Tailwind CSS y TypeScript, lo que permite una experiencia rápida, responsiva y altamente personalizable.

Backend: construido con Node.js y Express, encargado de manejar la lógica de negocio, autenticación, validaciones y conexión con la base de datos.

Agente IA: implementado como microservicio en Python usando LangChain y Groq. Este agente interpreta comandos, divide proyectos, asigna tareas y razona sobre el estado del sistema.

Base de datos: gestionada con Supabase (PostgreSQL) o directamente PostgreSQL, que almacena usuarios, proyectos, tareas, historial y relaciones entre entidades.

2. Comuncionacion entre componentes

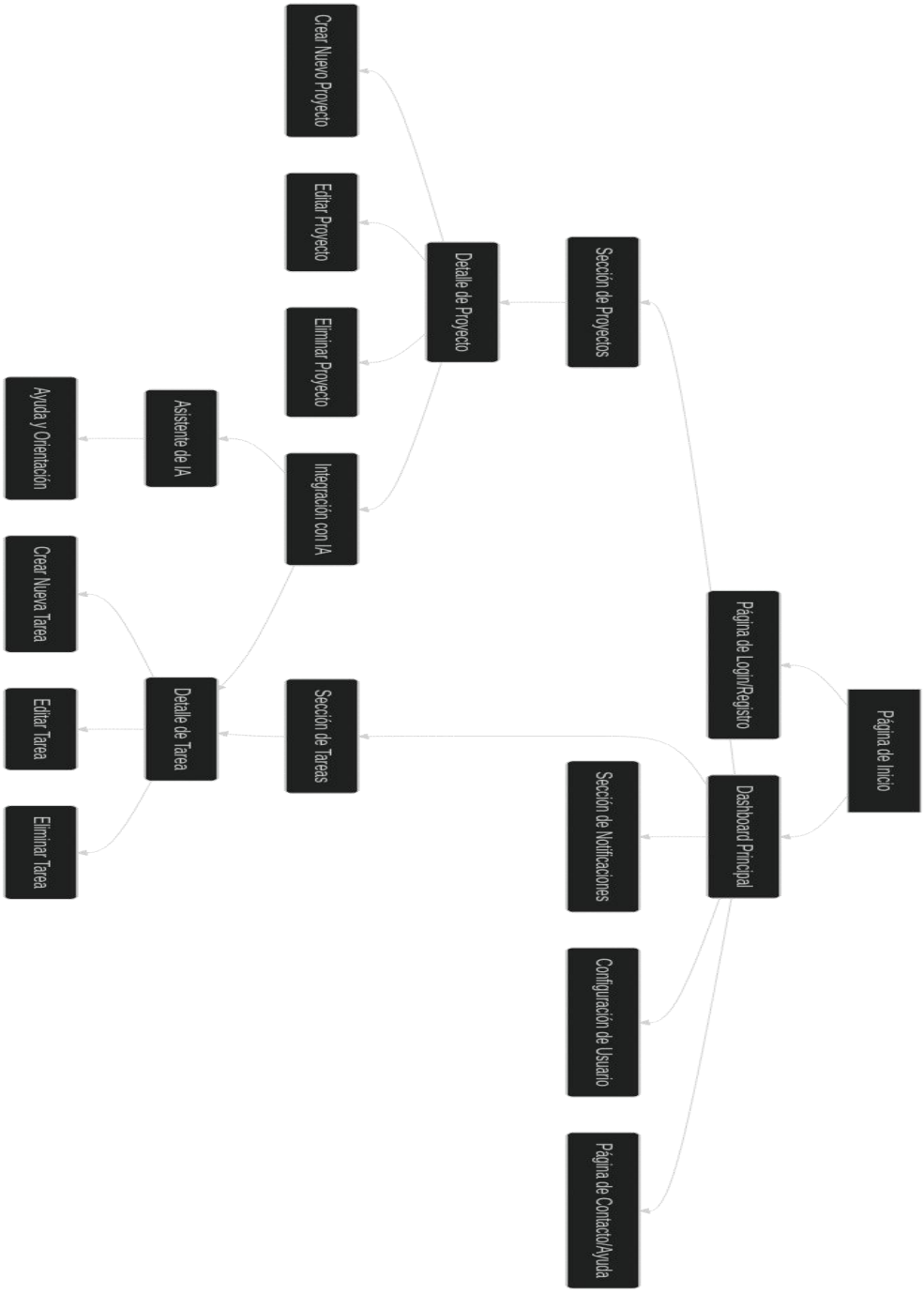
El frontend se comunica con el backend mediante API REST, enviando comandos, consultas y actualizaciones.

El agente IA se integra como microservicio separado, con posibilidad de ser embebido directamente en el backend si la carga lo requiere.

Las respuestas del agente se presentan en el frontend mediante un modal conversacional y un chat de proyecto, permitiendo interacción directa y contextual.

La base de datos se consulta desde el backend y el agente IA, permitiendo razonamiento basado en datos reales.

Diagrama de Arquitectura



Diseño de la Escalabilidad

Worklyst está pensado para crecer con el equipo que lo utiliza. Aunque el entorno inicial es de prueba con aproximadamente 50 usuarios activos, la arquitectura permite escalar sin comprometer rendimiento ni claridad visual.

1. Estrategias de Escalabilidad

- **Microservicios desacoplados:** el agente IA puede escalar de forma independiente, permitiendo balanceo de carga y procesamiento paralelo.
- **Escalabilidad horizontal:** el backend y el agente IA pueden replicarse en múltiples instancias si la demanda crece.
- **Componentes desacoplados en frontend:** cada vista (Dashboard, Proyecto, Modal IA) puede cargarse de forma independiente, facilitando lazy loading.
- **Renderizado eficiente:** se prioriza la carga de componentes clave y se minimiza el uso de efectos innecesarios.

2. Optimizacion visual tecnica

Lazy loading se aplicará en componentes e imágenes, especialmente en vistas como Dashboard y Modal IA.

Paginación se implementará en secciones con alto volumen de datos como proyectos e historial, evitando saturación de la interfaz.

Cache selectivo: se utilizará Supabase para cachear consultas frecuentes y localStorage para persistencia ligera en el navegador (por ejemplo, preferencias de usuario o estado de sesión).

Diseño de la Interfaz de Usuario

Principios de Diseño

La interfaz de Worklyst se basa en tres pilares: claridad visual, accesibilidad funcional y coherencia emocional. Cada vista está pensada para facilitar la lectura, reducir la carga cognitiva y adaptarse al modo de trabajo del usuario.

Jerarquía visual y navegación

La navegación se organiza en torno a cuatro vistas principales: Home, Dashboard, Proyecto y Modal IA.

- **Inicio** presenta una bienvenida contextual y accesos rápidos.
- **Dashboard** muestra el estado general de los proyectos y tareas.
- **Proyecto** permite visualizar y editar tareas, miembros y columnas.
- **Chat** Permite a los usuarios comunicarse mediante canales de proyecto
- **Modal IA** actúa como interfaz conversacional con el agente inteligente.

La jerarquía visual prioriza la información relevante, con componentes desacoplados que permiten personalización y escalabilidad.

Estilo Visual

La interfaz de Worklyst utiliza una estética funcional y emocionalmente coherente, pensada para facilitar la lectura, destacar prioridades y adaptarse a distintos contextos de uso. El sistema de diseño se apoya en una paleta de colores suaves con acentos estratégicos, y una tipografía moderna que refuerza la claridad y la personalidad de la plataforma.

Tipografía

La fuente principal utilizada en Worklyst es Poppins, una tipografía sans-serif moderna que combina legibilidad con personalidad. Su estructura geométrica y versatilidad la hacen ideal para interfaces limpias, jerarquizadas y adaptables a distintos tamaños de pantalla.

Paleta de colores

La paleta se compone de tonos neutros para estructura y fondo, y colores de acento para representar estados, prioridades y acciones clave. A continuación, se detallan los colores principales junto con sus equivalentes en Tailwind CSS:

Color Base	Hexadecimal	Clase Tailwind CSS
Blanco	#FFFFFF	white
Gris Claro	#F5F5F5	gray-100
Gris Oscuro	#374151	gray-700
Azul	#3B82F6	blue-500
Verde	#10B9B1	green-500
Rojo	#EF4444	red-500
Amarillo	#FBBF24	yellow-400

Entidades de Base de Datos

- **Usuarios :** Su clave primaria es ID_Usuarios Y se relaciona con la Entidad Rol Ya que varios usuarios pueden tener 1 Solo rol

M : 1

- **Rol:** Su clave primaria es ID_ROL

- **Sesion:** Su clave primaria es ID_Sesion Se relaciona con la entidad Usuarios ya que 1 usuario puede iniciar sesión varias veces.

M:1

- **Perfil_Habilidades:** Esta se relaciona con la entidad Usuarios ya que 1 usuario puede tener varias habilidades.

1:M

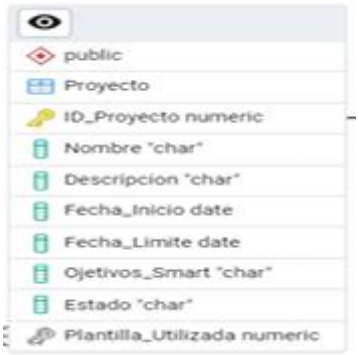
- **Historial_Usuario:** Esta entidad se relaciona con múltiples entidades, Usuarios, Tarea, Proyecto, Historial_Cambios, Chat_Bot_IA, Sesion, ya que se encarga de almacenar muchos datos de cada entidad.

M:1

- **Notificación:** Su clave primaria es ID_Notificacion Se relaciona con la entidad Usuario ya que 1 usuario puede tener varias notificaciones

1:M

- **Proyecto:** Su clave primaria es ID_Proyecto Se relaciona con la entidad Plantilla ya que 1 proyecto pueden tener varias plantillas.



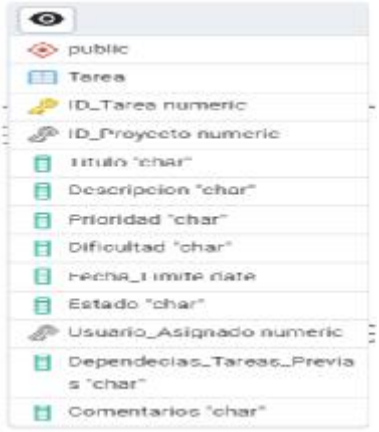
1:M

- **Plantilla:** Su clave primaria es ID_plantilla

- **Tablero:** Su clave primaria es ID_Tablero Se relaciona con la entidad Proyecto Ya que 1 proyecto puede tener varios Tableros.

M:1

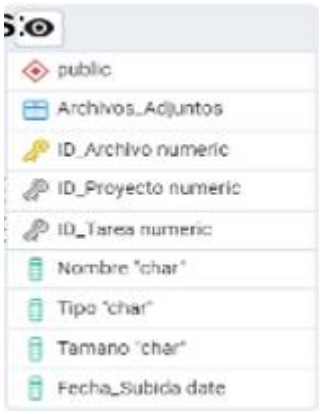
- **Tarea:** Su clave primaria es ID_Tarea Se relaciona con la entidad Proyecto y Usuarios ya que cada Usuario y cada Proyecto puede tener varias tareas.



- **Chat_Bot_IA:** Su clave primaria es ID_Conversacion Se relaciona con la entidad Usuarios Ya que cada usuario puede tener varias conversaciones con la IA.

M:1

- **Archivos_Adjuntos:** Su llave primaria es ID_Archivo Se relaciona con la entidad Proyecto y Tareas ya que cada Proyecto y Tarea puede tener varios archivos.

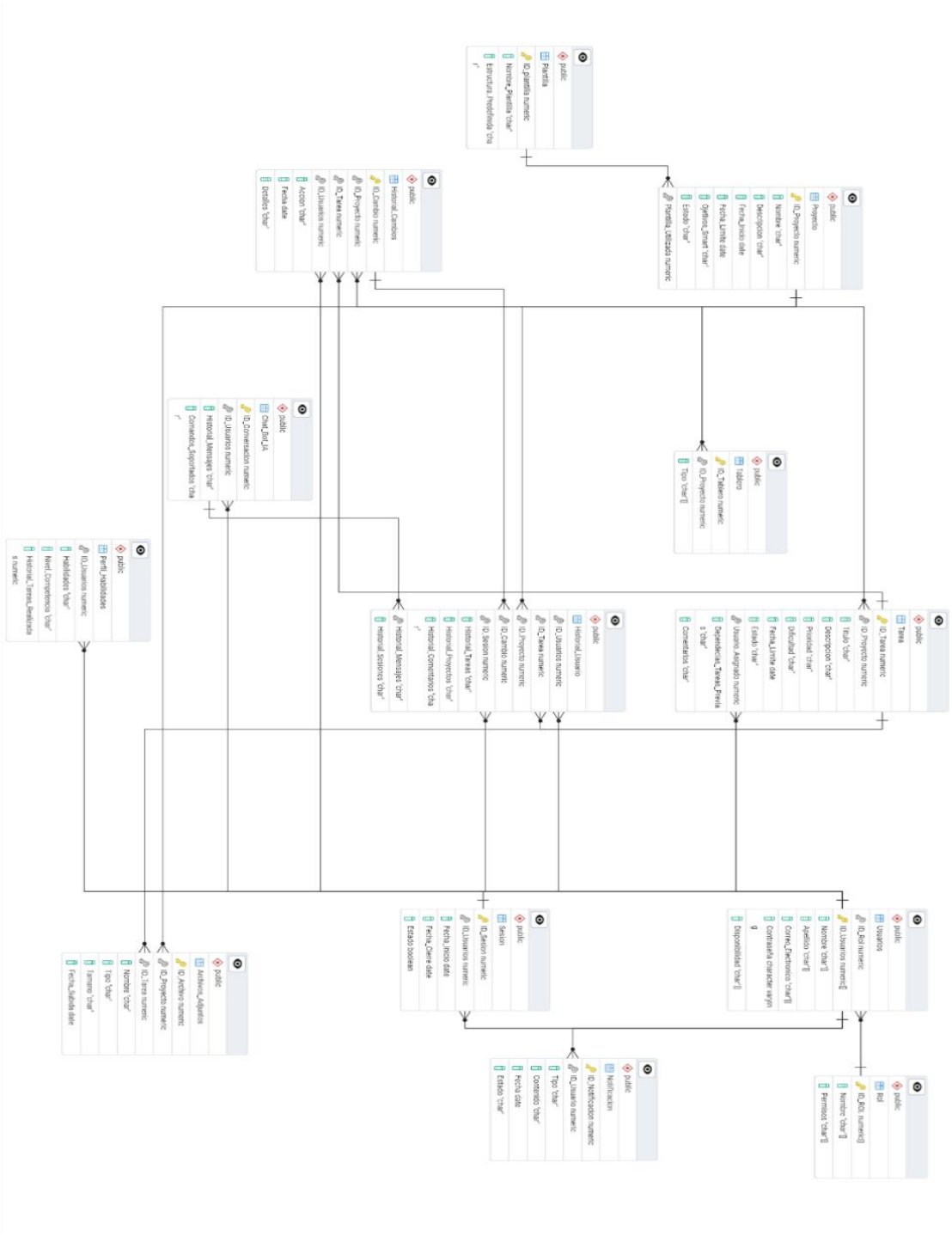


1:M

- **Historial_Cambios:** Su llave primaria es ID_Cambio Se relaciona con las entidades Proyecto, Tarea, Usuarios ya que cada proyecto, tarea, y Usuario está sujeto a múltiples cambios.

1:M

Diagrama Entidad Relación



Definición de autenticación y autorización

1. Autenticación

La autenticación es el proceso de verificar la identidad de un usuario antes de que pueda acceder a las funcionalidades del sistema. Este proceso garantiza que solo las personas legítimas puedan iniciar sesión

- **Método de Identificación:** El sistema requiere que los usuarios se identifiquen con un correo electrónico y una contraseña encriptada.
- **Requisitos Funcionales Clave:**
  - **RF 001 - Registro y autenticación de usuarios:** Es el requisito fundamental que permite a los usuarios acceder a la aplicación.
  - **RF 005 - Sistema de validación de credenciales:** Asegura que la información de inicio de sesión sea correcta y refuerza la seguridad del acceso.

2. Autorización y Roles

La autorización se refiere al control de acceso que determina qué acciones puede realizar un usuario una vez que ha sido autenticado. Se implementa mediante la asignación de roles, que son un conjunto de permisos predefinidos.

- **RF 003 - Roles y permisos:** Este requisito es de alta prioridad y establece la base del control de acceso para todas las funcionalidades del sistema.
- **Roles Definidos:**
  - **Administrador del Sistema:** Tiene la máxima autoridad, con permisos para gestionar roles, usuarios y la estabilidad general del sistema.
  - **Líder del Proyecto:** Encargado de la creación, configuración y gestión de colaboradores dentro de proyectos específicos.
  - **Usuario Regular:** Puede registrarse, editar su perfil y participar en proyectos a los que sea invitado.
  - **Visitante:** Un usuario no registrado con acceso limitado, principalmente para ver contenido público.

3. Gestion de Sesiones

Una sesión es el estado de la conexión de un usuario con la aplicación. La gestión de sesiones es vital para la seguridad, ya que asegura que la interacción del usuario esté controlada y protegida contra accesos no autorizados.

- **RF 006 - Gestión de sesiones:** Un requisito de alta prioridad que garantiza la seguridad y el control de las sesiones.
- **Atributos de la Sesión:** De acuerdo con el diagrama de clases, una sesión se define por su ID, el ID del usuario, la fecha y hora de inicio y la fecha y hora de cierre.

- **Seguridad:** Las sesiones deben estar protegidas y el sistema debe ser capaz de manejar conflictos, como ediciones simultáneas, para garantizar la integridad de los datos.

Identificación de Vulnerabilidades Comunes

1. Cross-Site Scripting (XSS)

El XSS es una vulnerabilidad que ocurre cuando un atacante inyecta código malicioso en una página web. Este código se ejecuta en el navegador de otros usuarios, lo que puede robar sus datos, como credenciales de sesión.

- **Riesgo en el proyecto:** El sistema de comentarios, los chats grupales y los campos de entrada de texto son puntos potenciales de ataque. Si estos campos no se validan y sanean adecuadamente, un atacante podría insertar un script malicioso que robe la sesión de un Líder de Proyecto o Administrador.
- **Mitigación:**
  - **Validación de entradas:** Implementar un filtro que elimine o escape caracteres peligrosos (<, >, &, ", ').
  - **Codificación de salida:** Asegurarse de que cualquier dato proveniente de los usuarios se muestre como texto plano y no como código ejecutable.

2. Cross-Site Request Forgery (CSRF)

El CSRF es un ataque que obliga a un usuario autenticado a realizar acciones no deseadas en una aplicación web en la que ha iniciado sesión. El atacante engaña al usuario para que envíe una solicitud a 5 la aplicación, como cambiar su contraseña o realizar una acción de borrado, sin su conocimiento.

- **Riesgo en el proyecto:** Las acciones críticas como crear un proyecto, eliminar una tarea, cambiar la contraseña o modificar permisos de un colaborador son vulnerables. Un atacante podría, por ejemplo, enviar un correo electrónico con un enlace que, al hacer clic, cambie el rol de un usuario sin que se dé cuenta.
- **Mitigación:**
  - **Tokens CSRF:** Implementar tokens únicos y aleatorios que se envíen con cada solicitud de formulario. El servidor verifica el token para asegurarse de que la solicitud es legítima y proviene del usuario.
  - **Verificación de encabezados:** Comprobar el encabezado Referer o Origin en las solicitudes para confirmar que provienen del mismo dominio de la aplicación.

3. Inyección de Código (SQL, NoSQL, etc.)

Las inyecciones de código ocurren cuando un atacante introduce datos maliciosos en un campo de entrada para manipular las consultas de la base de datos o el sistema operativo subyacente. La inyección de SQL es la más común y permite a los atacantes acceder, modificar o eliminar datos de la base de datos.

- **Riesgo en el proyecto:** La aplicación utiliza una base de datos para la gestión de usuarios, proyectos y tareas. Campos de búsqueda, formularios de inicio de sesión o cualquier campo que interactúe directamente con la base de datos son vulnerables. Un atacante podría, por ejemplo, usar una inyección de SQL para

eludir el inicio de sesión o acceder a la información de todos los usuarios.

- **Mitigación:**
  - **Consultas parametrizadas (Prepared Statements):** Esta es la defensa más efectiva. Las consultas se construyen de manera que los datos de entrada se envían por separado de la lógica de la consulta, evitando que se interpreten como comandos.
  - **Validación de entradas:** Limpiar y validar todos los datos de entrada del usuario para asegurar que no contengan caracteres especiales o comandos de base de datos.
  - **Principios de mínimo privilegio:** Limitar los permisos de la base de datos para que la aplicación no pueda realizar acciones no autorizadas, incluso si ocurre una inyección.

4. Gestión de Sesiones Insegura

Esta vulnerabilidad surge cuando el manejo de las sesiones de usuario no es robusto, lo que permite a un atacante robar, secuestrar o predecir los identificadores de sesión.

- **Riesgo en el proyecto:** Los documentos mencionan que la Gestión de Sesiones es un requisito de alta prioridad y que las sesiones deben estar protegidas. Un fallo en esta implementación podría permitir a un atacante obtener un identificador de sesión y usarlo para suplantar la identidad de un usuario, como un Líder de Proyecto o incluso un Administrador. Esto le daría acceso total

a sus funcionalidades y datos sin necesidad de conocer su contraseña.

- **Mitigación:**
  - **IDs de sesión seguros:** Generar identificadores de sesión largos, aleatorios e impredecibles.
  - **Uso de HTTPS:** Cifrar la comunicación para evitar que los identificadores de sesión sean interceptados en el tráfico de red.
  - **Tiempo de vida limitado:** Establecer un tiempo de caducidad para las sesiones. Por ejemplo, cerrarlas automáticamente después de un período de inactividad.
  - **Renovación de sesión:** Cambiar el identificador de sesión después de un inicio de sesión exitoso o al elevar los privilegios del usuario.

Gestión del Proyecto y Requisitos

Esta sección documenta la planificación, la estructura organizacional y las especificaciones técnicas y funcionales que servirán como línea base para el desarrollo de la plataforma **Worklyst**.

Plan de Desarrollo de Software (PDS)

Metodología y Alcance

- **Metodología:** Se utilizará **Kanban** como enfoque principal para la gestión del flujo de trabajo, lo que permite visibilidad total de los avances entre los integrantes del equipo.
- **Duración Estimada:** El tiempo estimado para la culminación del proyecto es de **cuatro (4) meses**.
- **Propósito del Proyecto: Worklyst** es una plataforma web para la gestión colaborativa de proyectos, enfocada en optimizar tareas y asignación de responsabilidades mediante la integración de un Agente de Inteligencia Artificial (IA).

Estructura Organizacional

El equipo opera bajo una estructura horizontal, donde todas las decisiones se toman por consenso.

Rol/Enfoque	Miembros del equipo
Frontend	Orlando López y Pedro Castro
Backend	Wrallean Brito y Jean Diaz

Planes de Iteraciones

El desarrollo se organiza en ciclos definidos por la finalización de una característica específica (*feature*), adoptando un enfoque de desarrollo basado en características.

- **Definición de Iteración:** Una iteración se cierra al momento de que una *feature* específica (ej. Autenticación) se encuentra completamente lista y probada.
- **Duración de Interación:** El tiempo de duración de cada ciclo de trabajo varía entre 3 a 12 días.
- **Estimación Total:** Se ha planificado un estimado de 8 iteraciones para el desarrollo completo del proyecto.

Lista de Riesgos y Estrategia de Mitigación

El foco de riesgo principal se encuentra en la dependencia de las tecnologías de Inteligencia Artificial y sus proveedores de servicio.

Riesgo	Impacto Potencial	Estrategia de Mitigación (EM)
Falla en la Interpretación de Comandos	El Agente IA no entiende o ejecuta mal los comandos naturales, afectando la funcionalidad clave.	**EM1:** Implementar un sistema de *logging* y monitoreo continuo para entrenar y refinar los *prompts* del Agente IA (LangChain).
Interrupción de Servicio de Groq	Indisponibilidad del servicio de inferencia de la IA, lo que inhabilita las funciones principales de la plataforma.	**EM2:** Mantener la arquitectura desacoplada para permitir una migración o el *switch* rápido a un proveedor de IA alternativo.
Coste por Token Inesperado	El uso intensivo del Agente IA genera costos operativos por token superiores a lo presupuestado.	**EM3:** Monitorear estrictamente el uso de *tokens* y aplicar **Cache Selectivo** para respuestas o consultas frecuentes.

Glosario del Proyecto

- **Agente IA:** Microservicio implementado con Python, LangChain y Groq, responsable de la interpretación de comandos naturales, división y asignación inteligente de tareas.
- **Kanban:** Metodología de desarrollo utilizada por el equipo, enfocada en la visualización del flujo de trabajo y la limitación de trabajo en progreso.
- **Groq:** Proveedor de servicio o hardware utilizado para la inferencia de modelos de lenguaje, esencial para el alto rendimiento del Agente IA.
- **Feature:** Una característica o funcionalidad completa del sistema que define el cierre y la entrega de una iteración (ej. Autenticación de Usuarios).

Análisis y Diseño

Esta sección documenta la arquitectura, el diseño de base de datos, la interfaz de usuario y los componentes clave que conforman la plataforma Worklyst. Su propósito es establecer la línea técnica que guiará la implementación y asegurar que las decisiones de diseño estén alineadas con los requisitos funcionales y no funcionales definidos anteriormente en este documento.

Especificación de Arquitectura de Software (EAS)

Metodología y Alcance La arquitectura se define bajo un enfoque modular y escalable, garantizando separación de responsabilidades y facilidad de mantenimiento.

- **Frontend:** React/Next.js con TypeScript, asegurando una interfaz responsiva y dinámica.
- **Backend:** Node.js con Express y TypeScript, encargado de la lógica de negocio y la comunicación con la base de datos.
- **Base de Datos:** PostgreSQL gestionada con Supabase, que provee autenticación, almacenamiento y consultas SQL.
- **Automatización e IA:** n8n como orquestador de flujos y Groq/OpenRouter como proveedores de modelos de lenguaje para el Agente IA.
- **Arquitectura General:** Cliente-Servidor con integración de microservicios para IA y automatización.

Diseño de Base de Datos

Esquema Detallado La base de datos se organiza en tablas principales:

- **users:** información de usuarios, roles y credenciales.
- **projects:** proyectos creados, con nombre, descripción y fecha de creación.

- **participants:** relación entre usuarios y proyectos.
- **tasks:** tareas asociadas a proyectos, con estado y asignación.

Relaciones

- Un proyecto tiene múltiples participantes.
- Un proyecto tiene múltiples tareas.
- Cada tarea se asigna a un participante.

Diccionario de Datos (ejemplo):

- projects.id → Identificador único del proyecto.
- tasks.status → Estado de la tarea (Pendiente, En Progreso, Completa).
- participants.user_id → Usuario asignado al proyecto.

Especificación de la Interfaz de Usuario (UI/UX)

Wireframes y Prototipos

- **Pantalla de inicio:** lista de proyectos activos.
- **Vista de proyecto:** tablero Kanban con tareas en columnas, soportando drag & drop.
- **Chat con Agente IA:** interfaz conversacional para dar instrucciones en lenguaje natural.

Guías de Estilo

- Tipografía clara y moderna.
- Paleta de colores neutros con acentos en azul/verde.
- Iconografía consistente y minimalista.

Documentación de Componentes

La documentación de componentes se plantea de manera general y flexible, sin imponer una estructura rígida.

- Se describen los módulos principales del sistema (frontend, backend, IA, automatización) indicando su propósito y cómo interactúan entre sí.
- Se incluyen las responsabilidades generales de cada bloque (ej. interfaz de usuario, lógica de negocio, persistencia de datos, procesamiento de IA).
- Se detallan las interacciones clave entre componentes, sin necesidad de definir clases o funciones específicas.
- El nivel de detalle podrá ajustarse según el avance del proyecto, permitiendo que el equipo documente lo esencial sin quedar atado a un formato fijo.

Planificación de Pruebas y Despliegues

Estrategia de Pruebas.

La estrategia de pruebas para Worklyst busca asegurar la calidad, escalabilidad y experiencia de usuario en una plataforma colaborativa impulsada por IA. Se prioriza la cobertura funcional, la robustez ante el crecimiento y la facilidad de evolución del sistema, alineando las pruebas con los objetivos de crear, gestionar y colaborar en proyectos de manera flexible y eficiente.

Enfoque y Niveles de Prueba.

El enfoque será ágil y modular, alineado con la arquitectura desacoplada de Worklyst. Se aplicarán pruebas automatizadas y manuales en los siguientes niveles:

1. **Pruebas Unitarias:** Cada componente (frontend, backend, microservicio IA) tendrá pruebas unitarias para validar funciones y métodos individuales.

2. **Pruebas de Integración:** Se validará la interacción entre módulos (por ejemplo, frontend-backend, backend-IA, backend-base de datos).

3. **Pruebas de Sistema:** Se probará la plataforma como un todo, asegurando que los flujos principales (crear proyectos, asignar tareas, chat, drag & drop) funcionen correctamente.

4. **Pruebas de Aceptación:** Validación con usuarios clave para asegurar que la aplicación cumple los objetivos y expectativas funcionales.

Tipos de Pruebas

Se contemplan los siguientes tipos de pruebas para cubrir tanto aspectos funcionales como no funcionales:

- **Funcionales:**
  - Creación y gestión de proyectos y equipos
  - Generación y personalización de tableros Kanban
  - Asignación inteligente de tareas y roles
  - Interacción en chat grupal y con IA
  - Drag & drop de tareas
- **No Funcionales:**
  - Usabilidad: Claridad visual, accesibilidad y coherencia emocional
  - Rendimiento: Respuesta bajo carga, especialmente en vistas Dashboard y Proyecto
  - Seguridad: Autenticación, autorización y protección de datos
  - Compatibilidad: Diferentes navegadores y dispositivos
  - Escalabilidad: Pruebas de carga y stress para validar el crecimiento horizontal y microservicios

Entornos de Pruebas

Se definirán varios entornos para asegurar la confiabilidad y reproduccion de los resultados:

- **Desarrollo:** Entorno local para pruebas rápidas y debugging.
- **Integracion:** Entorno intermedio donde se despliegan ramas de desarrollo para pruebas de integración y automatizadas.
- **Calidad (QA):** Réplica del entorno de producción, con datos simulados, para pruebas de sistema, aceptación y carga.
- **Producción:** Monitoreo (monitoreo de errores).

Cada entorno debe contar con configuraciones similares a producción (servidores, base de datos Supabase, microservicios IA, etc.) para garantizar la validez de las pruebas .

Responsables de la Calidad

La responsabilidad de la calidad se divide en dos equipos principales:

- **Equipo Backend:**
  - Pruebas unitarias y de integración de API, lógica de negocio, microservicio IA y base de datos.
  - Pruebas de rendimiento y seguridad en endpoints y servicios.
- **Equipo Frontend:**
  - Pruebas unitarias y de integración de componentes visuales y flujos de usuario.
  - Pruebas de usabilidad, accesibilidad y compatibilidad.

- Validación de la experiencia conversacional y de interacción (chat, drag & drop, modal IA).

Ambos equipos colaboran en pruebas de sistema, aceptación y escalabilidad, asegurando la cobertura de extremo a extremo.

Casos de Pruebas

Casos de prueba basados en los requisitos.

1. Crear proyectos personalizados.

- **Caso de Prueba 1.1:** Crear un nuevo proyecto con nombre y descripción
  - **Pre condiciones:** Usuario autenticado en la plataforma
  - **Pasos:**
    - Acceder a la opción “Crear proyecto”
    - Ingresar nombre y descripción
    - Guardar el proyecto
  - **Datos de entrada:**
    - **Nombre:** “Proyecto Alpha”
    - **Descripción:** “Desarrollo de nueva funcionalidad”
    - **Resultado esperado:**
      - El proyecto aparece en la lista de proyectos del usuario con los datos ingresados

2. Formar equipos y añadir miembros.

- **Caso de Prueba 2.1:** Añadir miembros a un proyecto existente
  - **Pre condiciones:** Proyecto creado, usuario con permisos de edición
  - **Pasos:**

- Acceder al proyecto
- Seleccionar “Añadir miembro”
- Ingresar correo electrónico del nuevo miembro
- Confirmar adición

- **Datos de entrada:**
  - **Correo:** usuario@ejemplo.com
  - **Resultado esperado:**
    - El nuevo miembro aparece en la lista de miembros del proyecto y recibe notificación de invitación

3. Generar tableros tipo Kanban o con otras plantillas visuales

- **Caso de Prueba 3.1:** Crear un tablero Kanban para un proyecto
  - **Pre condiciones:** Proyecto existente
  - **Pasos:**
    - Acceder al proyecto
    - Seleccionar “Crear tablero”
    - Elegir plantilla “Kanban”
    - Confirmar creación
  - **Datos de entrada:**
    - **Plantilla:** Kanban
    - **Resultado esperado:**
      - El tablero Kanban se muestra con columnas predeterminadas (Por hacer, En progreso, Hecho)

4. Interactuar mediante chat grupal y chat con IA

- **Caso de Prueba 4.1:** Enviar mensaje en el chat grupal del proyecto
  - **Pre condiciones:** Usuario es miembro del proyecto
  - **Pasos:**
    - Acceder al chat grupal del proyecto
    - Escribir mensaje
    - Enviar
  - **Datos de entrada:**
    - **Mensaje:** “Hola equipo”
    - **Resultado esperado:**
      - El mensaje aparece en el chat grupal visible para todos los miembros
- **Caso de Prueba 4.2:** Consultar al chat con IA sobre el estado del proyecto
  - **Pre condiciones:** Proyecto con tareas creadas
  - **Pasos:**
    - Acceder al chat con IA
    - **Escribir:** “¿Cuántas tareas están en progreso?”
    - Enviar
      - **Datos de entrada:**
        - **Pregunta:** “¿Cuántas tareas están en progreso?”
        - **Resultado esperado:**

- La IA responde con el número de tareas en progreso

5. Asignar tareas y roles de forma inteligente.

- **Caso de Prueba 5.1:** Asignación automática de tareas por IA
  - **Pre condiciones:** Proyecto con miembros y tareas sin asignar
  - **Pasos:**
    - Solicitar a la IA: “Asigna las tareas automáticamente”
  - **Datos de entrada:**
    - **Comando:** “Asigna las tareas automáticamente”
    - **Resultado esperado:**
      - Las tareas se asignan a los miembros según sus habilidades y disponibilidad

6. Utilizar drag & drop para organizar tareas.

- **Caso de Prueba 6.1:** Mover tarea de una columna a otra usando drag & drop
  - **Pre condiciones:** Tablero Kanban con tareas
  - **Pasos:**
    - Seleccionar una tarea en la columna “Por hacer”
    - Arrastrar y soltar en la columna “En progreso”
  - **Datos de entrada:**
    - Tarea seleccionada
    - **Resultado esperado:**

- La tarea aparece en la columna “En progreso” y se actualiza su estado

Plan de Despliegue y Configuración

Pasos para pasar el sistema a los entornos de staging y producción.

- **Preparación de Infraestructura.**
  - **Seleccionar los entornos:** Definir subdominios o dominios separados para staging (ej. `staging.worklyst.com`) y producción (`worklyst.com`).
  - **Verificar recursos:** Asegúrar contar con suficiente espacio en el hosting y recursos para ambos entornos .
- **Configuración del Entorno de Staging.**
  - **Clonación del Proyecto:**
    - Duplicar el código fuente del frontend, backend y microservicio de IA en el entorno de staging.
    - Crear una base de datos independiente para staging, clonando la estructura y, si es necesario, datos de prueba desde producción .
  - **Configuración de Variables de Entorno:**
    - Ajustar las variables de entorno para apuntar a los servicios y bases de datos de staging (URLs, claves API, endpoints de IA, etc.).
  - **Desplegar de Componentes:**
    - Desplegar el frontend (Next.js) y backend (Node.js/Express) en servidores o servicios cloud configurados para staging.
    - Desplegar el microservicio de IA en su entorno correspondiente, asegurando que apunte a la base de datos de staging.
    - Configurar Supabase o PostgreSQL para el entorno de staging.

- **Configuración de Accesos y Seguridad:**
  - Limitar el acceso al entorno de staging (por ejemplo, mediante autenticación básica o IP allowlist).
  - Desactivar la indexación en buscadores para staging .
- **Pruebas y Validación:**
  - Realizar las pruebas funcionales y de integración en staging antes de cualquier despliegue a producción .

- **Despliegue al Entorno de Producción**
  - **Verificación Previa:**
    - Asegúrar que la versión a desplegar haya pasado todas las pruebas en staging y esté aprobada por el equipo .
  - **Despliegue Controlado:**
    - Realizar el despliegue del frontend, backend y microservicio de IA en los servidores de producción.
    - Apuntar las variables de entorno a los servicios y base de datos de producción.
    - Realizar migraciones de base de datos si es necesario, asegurando respaldo previo.
  - **Configuración de Seguridad y Monitorización:**
    - Habilitar HTTPS, configurar firewalls y monitorización de logs.
    - Asegurar que solo el equipo de operaciones pueda realizar despliegues o rollbacks en producción.
  - **Verificación Post-Despliegue:**

- Realizar pruebas de humo para validar que los servicios principales funcionan correctamente.
- Monitorear el sistema para detectar posibles incidencias.

- **Flujo de Actualización**
  - **Staging:** Todas las nuevas versiones y cambios se prueban primero en staging.
  - **Producción:** Solo versiones estables y validadas pasan a producción, siguiendo un proceso de aprobación y checklist de validación .

Requisitos de Infraestructura y entornos

- **Definición de Entornos de Prueba**
  - **Entorno de Desarrollo:** Espacio donde los desarrolladores compilan y prueban localmente sus cambios. Debe ser ágil y flexible, con requisitos mínimos de infraestructura para facilitar iteraciones rápidas.
  - **Entorno de Staging (QA):** Réplica lo más fiel posible del entorno de producción, incluyendo configuración, base de datos y servicios. Aquí se realizan pruebas funcionales, de integración y de usuario final para validar que el sistema cumple con los requisitos antes de pasar a producción .
  - **Entorno de Producción:** Donde el sistema está disponible para los usuarios finales. Debe contar con alta disponibilidad, seguridad y monitorización.
- **Requisitos de Infraestructura para Pruebas**
  - **Capacidad para replicar escenarios de producción:** El entorno de staging debe poder simular cargas, configuraciones y datos similares a producción para detectar problemas reales.
  - **Aislamiento:** Cada entorno debe estar aislado para evitar interferencias entre pruebas y garantizar resultados confiables.
  - **Automatización de configuración:** Uso de herramientas de Infraestructura como Código (IaC) como Terraform, Ansible o Kubernetes para automatizar la creación y configuración de entornos, reduciendo errores y tiempos de despliegue
  - **Recursos escalables:** Infraestructura que permita ajustar recursos (CPU, memoria, almacenamiento) según las necesidades de las pruebas, optimizando costos.

- **Seguridad y control de accesos:** Implementar controles para limitar accesos a entornos de prueba, especialmente staging, para proteger datos y evitar modificaciones no autorizadas .

- **Monitorización y logging:** Sistemas para registrar eventos, errores y métricas durante las pruebas, facilitando la detección y análisis de fallos.

- **Configuración de Software y Herramientas**

  - **Base de datos independiente:** Cada entorno debe tener su propia base de datos para evitar contaminación de datos y permitir pruebas con datos específicos.

  - **Versionamiento y despliegue controlado:** Integración con pipelines CI/CD que permitan desplegar versiones específicas en cada entorno, garantizando trazabilidad y reversión si es necesario .

  - **Herramientas de prueba:** Incorporar frameworks para pruebas unitarias, de integración, de carga y de aceptación, que se ejecuten automáticamente en los entornos correspondientes .

- **Procedimientos y Gestión:**

  - **Solicitud y documentación de requisitos:** Establecer procedimientos claros para que los equipos definan y soliciten los requisitos de los entornos de prueba, incluyendo configuraciones, herramientas y datos necesarios .

  - **Validación y cualificación de infraestructura:** Realizar comprobaciones para asegurar que la infraestructura instalada cumple con el diseño y requisitos, incluyendo pruebas de backup, restore y seguridad .

  - **Planificación de pruebas:** Definir etapas y tipos de pruebas a realizar en cada entorno, asegurando que se cubren desde pruebas unitarias hasta pruebas de integración y de usuario final .

Plan de Mantenimiento

I. Objetivos del plan

- Asegurar la continuidad operativa y la disponibilidad del sistema web.
- Mantener la seguridad e integridad de los datos (PostgreSQL/Supabase).
- Garantizar un tiempo de respuesta eficiente ante incidentes y peticiones de soporte.
- Implementar actualizaciones y mejoras de manera controlada y probada.

II. Estructura de Soporte y Responsabilidades

Rol/Equipo	Responsabilidades	Frecuencia de Revision
Coordinación de TI/Líder Técnico	Máximo Responsable del Sistema. Supervisión del plan, toma de decisiones críticas, gestión de recursos, coordinación con equipos de desarrollo externos de ser necesario	Semanal/Mensual
Equipo de Desarrollo/Mantenimiento	Mantenimiento Correctivo, Preventivo y Evolutivo. Gestión de repositorios Git.	Diario, Mensual
Equipo de Infraestructura/Servidores	Monitoreo de servidores (RAM, CPU, Disco), gestión de red y firewalls, gestión de copias de seguridad a nivel infraestructura.	Diario

Help Desk / Soporte Nivel 1	Primer contacto con el usuario. Registro, clasificación y escalado de incidencias y peticiones (tickets). Resolución de problemas sencillos (contraseñas, caché, etc.).	Diario
Usuarios Clave / Administradores Funcionales	Pruebas de nuevas versiones, validación de funcionalidad, formación a usuarios finales, elaboración de manuales de usuario.	Según necesidad / Lanzamiento

III. Procedimientos de Soporte

a) Gestion de incidentes y errores

Este proceso asegura que los errores reportados sean atendidos y resueltos de manera oportuna.

i. **Reporte:** El usuario final reporta un problema a la **Help Deks**, creando un ticket.

ii. **Clasificación (Nivel 1):** La Mesa de Ayuda evalúa la incidencia según:

- **Gravedad/Impacto:** Crítico, Alto, Medio, Bajo.
- **Tipo:** Error de código, Error de datos, Error de infraestructura, Petición de mejora.

iii. **Escalado a Nivel 2:** Si el problema es un bug o requiere cambios en el sistema:

- El Nivel 2 reproduce el error y lo diagnostica.
- Se crea un branch en el repositorio (Git) para la correccion.

iv. **Desarrollo y Pruebas:**

- El desarrollador aplica la solución.

- El Usuario Clave/Administrador Funcional realiza la prueba de aceptación (UAT) en un ambiente de staging o pruebas.

v. **Implementación y Cierre:**

- El Equipo de Infraestructura despliega la corrección a producción.
- Help Desk confirma con el usuario la resolución y cierra el ticket.

b) Mantenimiento Preventivo y Actualizaciones

Tarea	Rol Responsable	Periodicidad Sugerida	Componentes Afectados
Revisión de Logs y Errores	Equipo de Desarrollo/Mantenimiento	Diaria	React, Express, Base de Datos
Actualización de Dependencias	Equipo de Desarrollo/Mantenimiento	Mensual Trimestral	NPM/Yarn, Node.js
Parches de Seguridad	Equipo de Infraestructura y Desarrollo	Inmediata	Servidor, Frameworks, Supabase/PostgreSQL
Revisión de Copias de Seguridad	Equipo de Infraestructura o Lider Técnico	Semanal	Base de datos y archivos
Análisis de Rendimiento	Equipo de Desarrollo/Mantenimiento	Mensual	Tiempos de carga, optimización de consultas SQL/API

c) Gestión de Peticiones de Cambio

Para incorporar nuevas funcionalidades.

i. **Solicitud:** La necesidad de una nueva funcionalidad se registra.

ii. **Análisis y Priorización:** La Coordinación de TI y el Usuario Clave evalúan la viabilidad, el impacto y la prioridad de la petición, añadiéndola al backlog.

iii. **Desarrollo, Pruebas y Despliegue:** Se sigue un ciclo similar al de la corrección de errores, pero con un enfoque en la arquitectura y la modificación de esquema de la DB si es necesario. Siempre se deben probar en ambiente de staging antes de producción.

IV. Consideraciones Específicas de la Pila Tecnológica

- **React/Express:** El mantenimiento se centra en la actualización de librerías de npm para seguridad y rendimiento, y en refactorizar el código para adaptarse a nuevas versiones de Node.js.
- **PostgreSQL/Supabase:**
  - **Seguridad de Datos:** La responsabilidad de gestionar los Row Level Security de Supabase debe recaer en el Equipo de Desarrollo.
  - **Backups:** Si se usa Supabase, la gestión de backups es en gran medida automática, pero el Líder Técnico debe validar periódicamente la configuración y la capacidad de restauración. Si se usa PostgreSQL self-hosted, el Equipo de Infraestructura es responsable de la política de copias de seguridad.