



REDDER

Proyecto de Grado Ingeniería en Computación

Autores

Martín Santini martinsantini1994@gmail.com
Lucas Suburú lucas.suburuch@gmail.com
Sebastian Zarrillo seba.zarrillo@gmail.com

Tutora

Libertad Tansini libertad@fing.edu.uy

Agradecimientos

Queremos expresar nuestro agradecimiento a las co-fundadoras de REDDER Global (Mariana Chango, Mercè Brey, Virginia Ceveri, Chus Sanz, Ivelise Reinaldo y Maria del Carmen Soto) quienes comenzaron este proyecto con el fin de ayudar a mujeres a desarrollarse personal y profesionalmente en diversas áreas.

A nuestra tutora Libertad Tansini por sus consejos, ayuda, dedicación y compromiso con el proyecto.

A todos los compañeros, profesores, amigos y familiares que estuvieron presentes en cada momento de la carrera.

Resumen

REDDER es una organización que tiene como fin construir una Red de Referentes conformada por mujeres y hombres, que inspiren y promuevan el desarrollo y la igualdad de oportunidades para las mujeres. Desde la organización creen en la tecnología como una aliada para llegar a todos los rincones y a todas las mujeres de una forma más eficiente.

El objetivo principal de este proyecto es la creación de una plataforma que facilite la gestión, el seguimiento y la interacción de las mentorías que se brindan desde la organización. A partir de esto, se diseñó e implementó un Producto Mínimo Viable (MVP) de una plataforma web que permite formar duplas de mentoría “Referente-Redderer”, dar seguimiento a las actividades realizadas, fomentar la interacción entre los integrantes de la red y digitalizar tareas de gestión que realiza el directorio de la organización como aprobar el registro de mentores, terminar parejas de mentoría, ver la cantidad de parejas activas en la red, entre otras.

El proyecto se dividió en dos grandes etapas, una primer etapa que consistió en trabajar en conjunto con el equipo de REDDER en la definición y alcance del proyecto, relevando y refinando los requerimientos necesarios para una versión inicial de la plataforma. Y una segunda etapa en la que se trabajó en el diseño e implementación de la plataforma, en esta etapa se trabajó bajo metodologías ágiles con el fin de presentar avances y recibir retroalimentación de parte de la organización en etapas tempranas del desarrollo de la plataforma.

Luego de varios meses de trabajo en conjunto con la organización se logró la implementación y el despliegue del sistema en un entorno real, y hoy en día ya está siendo utilizado por mentores y mentoreadas en su día a día. Finalmente se valida la plataforma con las integrantes de la red, quienes evaluaron muy positivamente el proceso de desarrollo y el resultado final.

Índice

1. Introducción	12
1.1. Motivación	12
1.2. Objetivos	12
1.3. Organización del documento	13
2. Antecedentes y conceptos básicos	14
2.1. Organizaciones y plataformas similares	14
2.2. Conceptos básicos	16
2.2.1. Referente	17
2.2.2. Redderer o Referida	17
2.2.3. Área de Interés	17
2.2.4. Dupla Referente-Redderer	19
3. Relevamiento de requisitos	20
3.1. Proceso de descubrimiento	20
4. Requerimientos	23
4.1. Aplicación web (para usuarios finales)	23
4.1.1. Referidos al manejo de sesión	23
4.1.2. Referidos al perfil de los usuarios	27
4.1.3. Referidos al flujo de selección de Referente	29
4.1.4. Referidos al flujo de aceptación de Referidos	31
4.1.5. Referidos al seguimiento de objetivos y feedback	32
4.1.6. Referidos a finalización de una pareja	34
4.2. Aplicación de administración	36
4.2.1. Referidos al manejo de sesión	36
4.2.2. Referidos al manejo de usuarios de administración	38
4.2.3. Referidos al manejo de la aplicación	40
4.3. Requerimientos no funcionales	46
4.3.1. Concurrencia	46
4.3.2. Usabilidad	46
4.3.3. Mantenibilidad	47
4.3.4. Escalabilidad	47
4.3.5. Web responsiva	48
4.3.6. Seguridad y privacidad de los datos	48

5.	Diseño de la solución	49
5.1.	Arquitectura del sistema	49
5.2.	Diseño base de datos	51
6.	Implementación de la solución	54
6.1.	Descripción general	54
6.1.1.	Aplicaciones web para usuarios finales	56
6.1.2.	Backoffice o Aplicación de administración	57
6.1.3.	Base de datos	57
6.1.4.	API o Backend	58
6.2.	Tareas programadas	59
6.3.	Integraciones	60
6.3.1.	Autenticación de Google y LinkedIn	60
6.3.2.	Sendgrid	61
6.3.3.	Azure	62
6.3.4.	Google Analytics	63
7.	Despliegue de la solución	65
7.1.	Netlify	65
7.2.	Heroku	65
7.3.	Azure	65
7.4.	MongoDB Atlas	67
8.	Pruebas y validaciones	69
8.1.	Revisión de código	69
8.2.	Pruebas funcionales	71
8.3.	Validación con usuarios	71
9.	Gestión del proyecto	74
9.1.	Metodología	74
9.1.1.	Metodologías Ágiles	74
9.1.2.	Control de versiones de código	77
9.2.	Herramientas	79
9.2.1.	Gestión del proyecto	79
9.2.2.	Comunicación	87
9.3.	Resumen	89
10.	Conclusiones y trabajo futuro	91

10.1. Conclusiones	91
10.2. Trabajo futuro	92
A. Documento sobre áreas de interés	98
B. Maquetas de Balsamiq	100
C. Encuesta realizada a las integrantes del directorio de la red	106
D. Endpoints Redder API	109
E. Los principios del Manifiesto Ágil	111
F. Modelo Entidad Relación	112
F.1. Usuario	112
F.2. Áreas de Interés	113
F.3. Tarea Programada	113
F.4. Pareja	114
F.5. Reporte	115
G. Pruebas de validación	116

Glosario

API *Application Programming Interface*, en español interfaz de programación de aplicaciones. Es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos o más aplicaciones de software a través de un conjunto de reglas.

API REST Es una API que sigue los lineamientos de la arquitectura REST (Representational State Transfer). Los lineamientos más importantes son:

- Cliente-servidor: las aplicaciones existentes en el servidor y el cliente deben estar separadas.
- Sin estado: las peticiones se realizan de forma independiente, es decir, cada petición debe autenticarse por sí sola y ejecuta solamente una acción determinada.

Backend (BE) Es la parte del sistema que el usuario no ve, se encarga de la lógica del mismo además de almacenar y comunicar los datos entre el frontend y la base de datos.

Backoffice (BO) Es la parte del sistema que se encarga de administrar el mismo. Por lo general tienen acceso únicamente integrantes del equipo de operaciones y no es una plataforma pública.

Business to business (B2B) Es un modelo de negocio en el cual una empresa comercializa sus productos o servicios a otra empresa.

Base de Datos (DB) Encargada de almacenar información y datos necesarios para el sistema.

Express Es un framework de desarrollo minimalista para NodeJS que permite estructurar una aplicación de manera ágil, proporcionando funcionalidades como el enrutamiento, opciones para gestionar sesiones entre otras.

Framework Es un *esquema o marco de trabajo* que ofrece una estructura base para elaborar un proyecto con objetivos específicos, una especie de esqueleto que sirve como punto de partida para la aplicación y el desarrollo de software.

Frontend (FE) Es la parte del sistema de acceso público con la cual interactúan los usuarios, por lo general es una interfaz gráfica.

Git Es una herramienta de control de versiones de código de forma distribuida y segura. Permite el manejo de código de diferentes usuarios de manera sencilla y rápida.

HTTP *HyperText Transfer Protocol* o, en español, Protocolo de Transferencia de Hiper Textos es un protocolo de transmisión de datos de la World Wide Web, habilitando de ésta manera la comunicación de diferentes computadores en la web.

HTTPS No es nada más que una versión segura del HTTP, es decir, una variante del mismo protocolo que se basa en la creación de un canal cifrado para la transmisión de la información, lo cual lo hace más apropiado para ciertos datos de tipo sensible. Hoy en día es el protocolo más normal a utilizar, debido a la capa de seguridad que le agrega a HTTP.

JSON *JavaScript Object Notation* o, en español, notación de objeto de JavaScript, es un formato de texto sencillo utilizado para el intercambio de datos.

JWT *JSON Web Token*. Es un estándar que define una manera segura de transmitir información a través de JSON. Lo que hace a JWT seguro es su firma digital, la cual permite que quien lo recibe pueda verificar su autenticidad y saber si ha sido modificado o no. Por este motivo es ampliamente utilizado para la autenticación en la aplicaciones web modernas.

MVP *Minimum Viable Product* o, en español, el *Producto Mínimo Viable* es la versión más sencilla de un producto y, por tanto, presenta sus funcionalidades más básicas de la aplicación a construir. Usualmente se utiliza para validar una idea.

MER *Modelo entidad-relación*. Herramienta para el modelado de datos, la cual facilita la representación de entidades de una base de datos y utilizada en etapas tempranas del desarrollo para el diseño de la misma (o para documentación de la DB en etapas más avanzadas).

MERN Es un conjunto o stack de tecnologías que se usan en conjunto, estas son MongoDB-Express-React-NodeJS.

MongoDB Es una base de datos no relacional orientada a documentos. Esto quiere decir que en lugar de guardar los datos en registros y tablas, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON.

NodeJS Es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor basado en el lenguaje de programación JavaScript que utiliza un modelo asíncrono y dirigido por eventos.

Normas ISO 25000 Conocida como SQuaRE (System and Software Quality Requirements and Evaluation), es una familia de normas que tiene por objetivo la creación de un marco de trabajo común para evaluar la calidad del producto de software.

ORM *Object Relational Mapping* o, en español, Mapeo Objeto-Relacional es una interfaz que permite convertir los datos de tus objetos de la aplicación a un formato correcto para poder guardar la información en una base de datos (mapeo).

ONG *Organización No Gubernamental*, institución sin ánimo de lucro que no depende del gobierno y realiza actividades de interés social.

Pull Request (PR) Son un mecanismo para que un desarrollador notifique a los miembros del equipo que ha completado una funcionalidad o parte de ella. Una vez que está lista, el desarrollador crea un pull request. Esto les permite a todos los involucrados saber que necesitan revisar el código ya que se muestran únicamente los cambios realizados al código base. Si hay algún problema con los cambios, los compañeros del equipo pueden publicar comentarios, dejando así la posibilidad de tener una calidad de código alta y uniforme a lo largo de la aplicación.

React Es una librería de JavaScript de código abierto para desarrollar interfaces de usuario.

REDDER Es una organización sin fines de lucro que tiene como objetivo el desarrollo de las mujeres y la equidad social a nivel global. Se apoya en la tecnología para llegar de forma más eficiente y rápida a todas las mujeres que necesiten de una mentoría a nivel profesional o personal.

Redderer Referido a un tipo de usuario de la plataforma REDDER. Este tipo de usuario va a ser mentoreado por un usuario Referente.

Referente Referido a un tipo de usuario de la plataforma REDDER. Este tipo de usuario va a ser el encargado de realizar las mentorías a los usuarios Redderers.

Smartphone En español, teléfono inteligente, es un celular que permite más que simplemente hacer llamadas, brinda la posibilidad de conectarse a internet, instalar aplicaciones o llevar a cabo muchas de las actividades que podrían realizarse en una computadora.

SPA Una *Single Page Application* o, en español, una aplicación de una página, es una web en la cual está todo el contenido en una sola página, es decir, carga tan solo un archivo HTML y todo se produce dentro de este único archivo. De esta manera se puede ofrecer una experiencia más fluida y más rápida.

1. Introducción

1.1. Motivación

Aunque las políticas de género llevan años en las agendas internacionales aún en pleno siglo XXI todavía muchas niñas y mujeres siguen sufriendo abuso de poder y se ven afectadas tanto en contextos laborales como personales por el simple hecho de ser mujeres. Si bien se han visto avances en el último tiempo seguimos viviendo en una sociedad desigual en cuanto a oportunidades, donde todavía hay mucho por aprender y mejorar para ser una sociedad más justa donde realmente exista igualdad y equidad de género. [1] [2]

En este contexto surge REDDER Global, una organización conformada por hombres y mujeres que tiene como fin el apoyo al desarrollo profesional de las mujeres y la equidad social a nivel global. Desde la organización creen que este desafío requiere de todas las personas, hombres y mujeres de forma inclusiva y complementaria. A su vez creen que la tecnología es una aliada para llegar a todos los rincones y a todas las mujeres con mentorías específicas y personalizadas para las necesidades de cada una. [3]

1.2. Objetivos

El objetivo principal de este proyecto es diseñar e implementar una plataforma que facilite y promueva la interacción y el aprendizaje en las mentorías que brindarán las referentes de la red a las “redderers”. La plataforma servirá en primer instancia para conformar parejas entre mentores y mentoreadas que compartan áreas de interés y luego brindar facilidades para el seguimiento de los objetivos de las parejas ya conformadas. Algunas de estas facilidades son el registro de las reuniones, agregar y actualizar objetivos y también agregar notas que sean de ayuda para la pareja.

Una vez finalizada la pareja las mujeres que recibieron la mentoría podrán dejar retroalimentación sobre los mentores con el fin de que el equipo de REDDER pueda mejorar el proceso de las mentorías.

A su vez, para ayudar al equipo de operaciones de REDDER en la gestión diaria de la plataforma y brindar un entorno seguro se desarrolla una aplicación web que servirá de backoffice. En esta aplicación se podrán procesar las solicitudes de registro de los referentes, ver denuncias de los

usuarios, cancelar parejas si se detectan actividades irregulares y ver información sobre las parejas conformadas entre varios otros casos de uso que se describen más adelante.

1.3. Organización del documento

Este informe se encuentra dividido en diez capítulos incluyendo este capítulo introductorio. En el [capítulo 2](#) se presentan los antecedentes del proyecto listando algunas plataformas similares y se detallan algunos de los conceptos teóricos más básicos para la comprensión de este informe y de la plataforma desarrollada. El [capítulo 3](#) describe la primer etapa del proyecto, el proceso de relevamiento y refinamiento de requisitos durante el cual se trabajó en conjunto con el equipo de REDDER. En el [capítulo 4](#) se listan los casos de uso para la plataforma de administración y de los usuarios finales así como también los requerimientos no funcionales del sistema.

En el [capítulo 5](#) se presenta el diseño de la solución a construir, describiendo la arquitectura de la misma, mientras que en el [capítulo 6](#) se presenta la implementación de la solución y los diversos componentes que la conforman. El [capítulo 7](#) está destinado a describir como fue el proceso de despliegue de las aplicaciones en los diversos entornos y las herramientas utilizadas durante este proceso, tanto en una etapa temprana del desarrollo como el despliegue final.

Por otro lado, en el [capítulo 8](#) se describen las tareas relacionadas a la evaluación y validación del software. En el [capítulo 9](#) se presentan las metodologías de gestión de proyectos y las diversas herramientas que utilizó el equipo durante el proyecto para la gestión del mismo.

Finalmente en el [capítulo 10](#) se comentan las conclusiones generales del proyecto realizado y se proponen posibles trabajos a futuro para seguir mejorando el sistema implementado.

Además, al final de este informe se cuentan con algunos anexos que poseen información complementaria al resto de los capítulos.

2. Antecedentes y conceptos básicos

En este capítulo se presentan los antecedentes del proyecto y los conceptos teóricos más básicos para la comprensión de este informe y del sistema desarrollado.

2.1. Organizaciones y plataformas similares

En los primeros encuentros con el directorio de la red nos transmitieron la visión y misión de la organización y como la tecnología por medio de una plataforma web podría ayudarlas. Tenían claro como la plataforma las ayudaría en sus operaciones del día a día pero aún había mucho trabajo por hacer de definición para refinar los casos de uso y requisitos de la plataforma.

En este proceso se investigan distintas plataformas de mentorías en línea y organizaciones con objetivos similares.

GrowthMentor

Es una plataforma de mentores relacionados con marketing en la cual los mismos solamente se pueden registrar como mentores mediante invitación de la plataforma. Estas invitaciones son enviadas únicamente a referentes con experiencia comprobada en sus respectivos campos. [4]

Emprendedores y especialistas en marketing pueden registrarse y solicitar mentorías con estos especialistas los cuales brindaran sesiones uno a uno a través de herramientas conocidas de videoconferencias con el fin de ayudar a emprendedores que tienen que tomar decisiones o validar ideas en soledad.

Al comparar esta plataforma con las ideas que tenían desde el directorio de REDDER en un comienzo se encontraron muchas similitudes, como pueden ser el control de quien es referente o mentor así como la idea de qué el acompañamiento es algo personal, uno a uno.

MicroMentor

Es un servicio de mentoría gratuito para emprendedores impulsado por mentores voluntarios. Es una innovación social de MercyCorps, una organización global líder en ayuda humanitaria que recibe financiamiento

de patrocinadores y donantes. [5]

En esta plataforma tanto los emprendedores como los mentores se pueden registrar y enviar solicitudes de mensajes o emparejamiento a los otros usuarios.

A diferencia de GrowthMentor y de forma similar a REDDER esta plataforma es gratuita para todos los usuarios.

MentorCloud

Es una plataforma B2B en la nube que proporciona a las organizaciones la posibilidad de integrar herramientas de tutoría en sus procesos.

Algunas de las compañías que utilizan esta plataforma son Airbnb, Marriot, Groupon, Expedia entre otras. [6]

Esta plataforma presenta varias diferencias con lo que plantea REDDER ya que cuenta con sus propias herramientas de chat y videoconferencias y además esta orientado a organizaciones y no a usuarios finales.

RedME - Red de Mujeres Ejecutivas del Uruguay

Como su nombre lo indica, es una red de mujeres ejecutivas que promueve el liderazgo de las mujeres para transformar las organizaciones del Uruguay desde la equidad y complementariedad. [7]

Esta organización propone acercar a mujeres que así lo desean a una mentora la cual, a través de su conocimiento y experiencia personal, sea una guía para lograr alcanzar los objetivos planteados. No posee una plataforma digital para esto, sino que se hace de forma manual, y, para registrarse en esta propuesta se debe completar un formulario y abonar una membresía anual.

Esta organización tiene muchas similitudes con la visión y casos de uso de REDDER, pero no presenta una forma automatizada y digital para sus procesos.

HLF - Hace la fuerza

Esta plataforma es una comunidad de mentoría para mujeres de Latinoamérica que tiene como misión cerrar la brecha de género en el ámbito profesional. [8]

A diferencia de alguna de las plataformas mencionadas anteriormente en

esta hay varios requisitos para aplicar para una mentoría. Estos requisitos son: ser mujer, ciudadana de Argentina, Colombia, México, Paraguay, Perú o Uruguay, estudiante avanzada o reciente egresada de una carrera de grado y ser primera generación universitaria de la familia. Una de las similitudes con REDDER es que las mentoreadas deben ser mujeres.

Plataforma / Organización	Costo	Accesibilidad	Plataforma	Idiomas	Area	Año Fundación
GrowthMentor	120 usd x mes	Todos	Plataforma propia	Inglés	Marketing	2017
MicroMentor	Gratis	Todos	Plataforma propia	Inglés, Español, Árabe, Indonesio, Somalí	Todas	2001
MentorCloud	N/A (B2B)	Empresas	Plataforma propia	Inglés	Todas	2013
RedMe	Gratis	Mujeres	No	Español	Todas	2020
HLF (Hace La Fuerza)	Gratis	Mujeres	Externa: MentorCloud	Español	Todas	2021
REDDER	Gratis	Mujeres	Plataforma propia	Español, Inglés	Todas	2021

Figura 1: Comparación de Plataformas / Organizaciones

En la *Figura 1* se puede observar un cuadro comparativo de las plataformas similares estudiadas. Las dos plataformas más similares a REDDER son RedMe y Hace La Fuerza, pero la principal diferencia que presentan es la plataforma ya que RedMe no cuenta con plataforma propia (sus procesos son manuales), y Hace La Fuera utiliza la solución para empresas MentorCloud. Por otro lado, GrowthMentor y MicroMentor tienen un público objetivo muy distinto a REDDER ya que GrowthMentor se enfoca en mentorías para el área de marketing, mientras que MicroMentor se enfoca en emprendedores.

2.2. Conceptos básicos

Hay varios conceptos que vale la pena explicar ya que tienen un significado específico dentro del contexto de REDDER.

2.2.1. Referente

Se le da el nombre de “Referente” a todas las personas que tienen experiencia comprobable en ciertas áreas y que tienen el deseo de ayudar a otras mujeres a desarrollar su potencial.

En las duplas que se conforman utilizando la plataforma las “Referentes” son las encargadas de guiar y ayudar a las “Redderers” brindando mentoría durante algunos meses.

Es posible que en ocasiones se haga referencia a una “Referente” con el sinónimo de “Mentora”.

Con el fin de brindar un entorno seguro para todos, antes de que una “Referente” forme parte de la plataforma hay un proceso de selección en el cual el directorio de la organización analiza las solicitudes de “Referentes” y aprueba o rechaza las mismas.

En el siguiente enlace se pueden ver algunas de las referentes de la plataforma: <https://www.redderglobal.org/referentes>

2.2.2. Redderer o Referida

“Redderer” es la persona que buscará “Referentes” para conformar una pareja y una vez conformada la pareja será quien recibirá ayuda profesional y seguimiento por parte de una “Referente”.

Las “Redderers” deberán registrarse desde la aplicación web ingresando sus datos personales y áreas de interés en las que desean desarrollarse por medio de mentorías de “Referentes” con experiencia comprobada en esas áreas.

Es posible que en ocasiones se haga referencia a una “Redderer” con el sinónimo de “Mentoreada” o “Referida”.

2.2.3. Área de Interés

En el contexto del sistema de REDDER las “Áreas de Interés” son un conjunto de especializaciones o temáticas en las cuales las “Referentes” y “Redderers” podrán brindar mentorías o ser mentoreadas.

Estos conjuntos no son exactamente los mismos para los “Referentes” y “Redderers”, a continuación se pueden ver estos conjuntos según el tipo de

usuario.

Áreas de Interés para Referentes

- Empresa
- Deporte y Salud
- Arte
- Ciencia y Tecnología
- Política
- Ámbito Rural
- Carrera Profesional

Áreas de Interés para Redderers

- Carrera Profesional
- Profesional Independiente
- Tengo una startup/pyme
- Carrera deportiva
- Arte
- Carrera científica
- Carrera política
- Soy mujer rural

Según las áreas de interés seleccionadas por ambos tipos de usuarios la plataforma recomendará ciertos perfiles de “Referentes” a las “Redderers”, quienes podrán enviar solicitudes para conformar duplas de trabajo a los “Referentes”. Estas recomendaciones se basan en la correspondencia que el equipo de la red definió entre las áreas de interés. Este documento se puede ver en el [anexo A](#).

2.2.4. Dupla Referente-Redderer

Una dupla estará conformada por una “Referente” y una “Redderer”, esta tendrá una duración máxima en meses que será definida por el directorio de REDDER (en principio tres meses). Durante ese tiempo la dupla tendrá distintos objetivos, reuniones y conversaciones las cuales podrá registrar y hacer seguimiento en la plataforma.

Mientras una “Referente” está conformando una pareja no podrá recibir más solicitudes de otros “Redderers”, esta es una de las razones por las cuales se decidió limitar la duración de las duplas de trabajo.

3. Relevamiento de requisitos

En este capítulo se describe la primer etapa del proyecto, el relevamiento de los requisitos.

El mismo se basa en un trabajo en conjunto con el equipo del directorio de REDDER con el fin de definir los casos de uso y conceptos para una primera versión (prototipo) de la plataforma. En esta etapa de relevamiento de requisitos se realizaron varias reuniones en las cuales se discutieron los posibles casos de uso que se debían incluir. Se utilizaron varios diseños/maquetas para ir validando los casos de uso en conjunto antes de comenzar con la implementación de los mismos.

3.1. Proceso de descubrimiento

La primera parte de este proceso consistió en una reunión inicial de todo el equipo de trabajo con el directorio de REDDER en donde se transmitió la idea de la organización y como se deseaba materializarla.

Como entregable de esta reunión inicial, se realizó un breve resumen de su idea por escrito, donde luego de correcciones y validación por parte de la organización, el resultado del mismo fue el siguiente:

“El objetivo principal consiste en posibilitar el madrinazgo entre Referente y Referido. Esto se planea hacer mediante un sistema de “matcheo”, donde se tendrán Referentes, que tendrán que ser validadas por la ONG y agregadas a través de un Backoffice, y Referidos, que se registraron en la aplicación web.

A los Referidos les aparecerán Referentes vinculados al ámbito o ámbitos que seleccionó al momento de la inscripción, pudiendo ver información relevante y seleccionando si le interesa (o no) la Referente en cuestión. Por el otro lado, la Referente tendrá una ‘Lista de solicitudes de Madrinazgo’, donde podrá ver todas las Referidas que se interesaron en ella, seleccionando una de ellas quedará la relación de Referencia establecida, que será por un tiempo establecido.”

Luego de estas primeras instancias en conjunto con el directorio en donde se transmitió cómo la tecnología, mediante esta plataforma, ayudaría a la organización con su misión, se comenzó con el proceso de crear los diseños de las maquetas para validar la idea previo a comenzar con la implementación

de las mismas.

El maquetado consta de una implementación parcial, la cual permite a los desarrolladores, usuarios y partes interesadas entender mejor los requisitos teniendo como ventaja acotar la incertidumbre de esa idea inicial y poder recibir retroalimentación en etapas tempranas. Para la creación de estos maquetados se utilizó una aplicación web llamada *Balsamiq*[9].

A continuación se presentan algunas de las maquetas más importantes que sirven de referencia para una gran cantidad de requerimientos y funcionalidades:

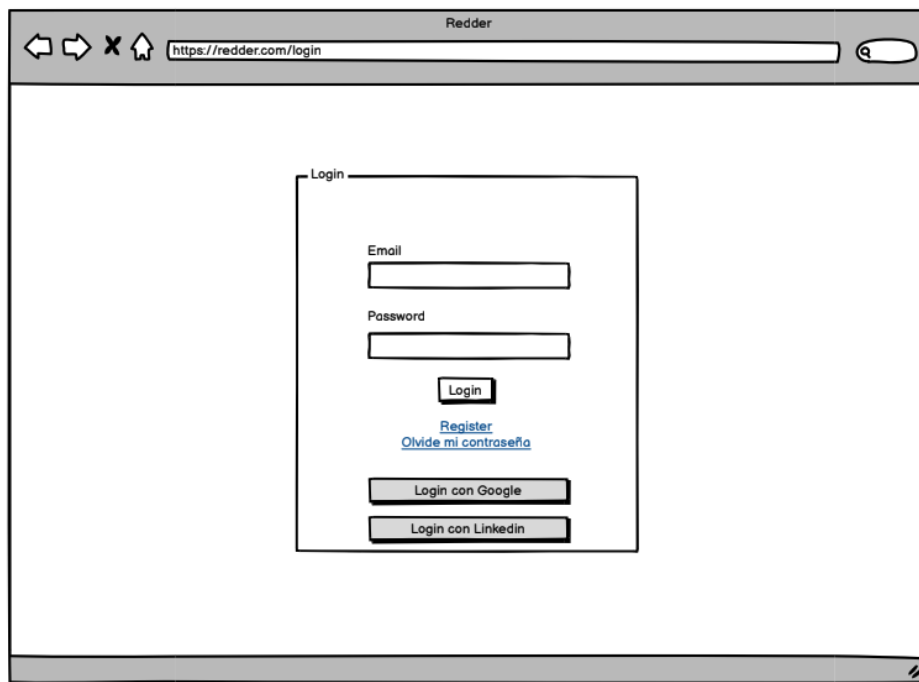


Figura 2: *Página de autenticación de usuarios*

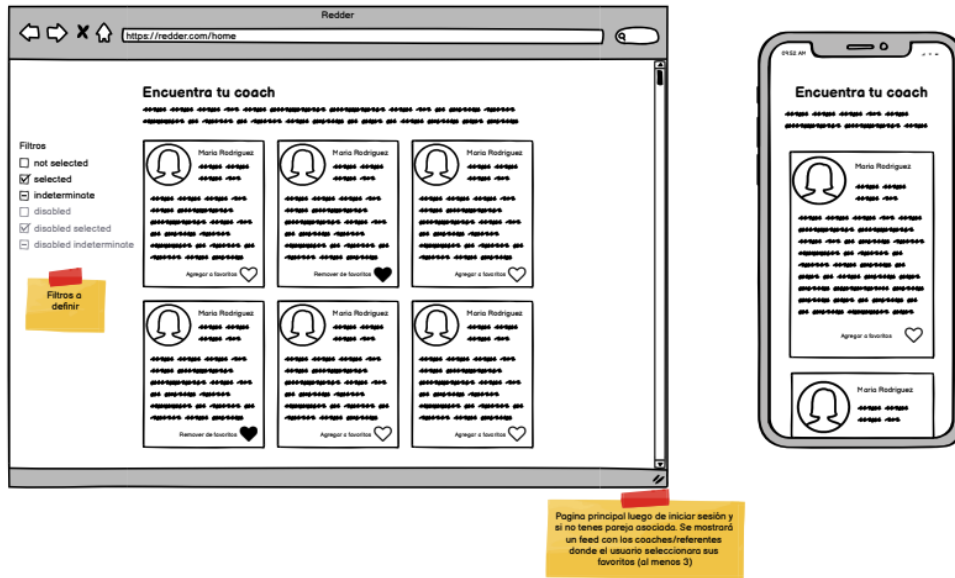


Figura 3: *Página de inicio de Redders sin pareja*

Como se puede observar en las figuras 2 y 3, se utilizaron estos maquetados intercambiando notas en los mismos con el equipo de REDDER logrando así validar y adaptar las ideas en una etapa temprana, antes de comenzar a desarrollar el proyecto. El resto de las maquetas se pueden encontrar en el Anexo B.

Luego de varias reuniones en donde se fueron refinando y adaptando estos diseños, se comenzó con la definición de requerimientos y casos de uso para la plataforma. Con este proceso se logró acotar el riesgo de desarrollar funcionalidades que no aportaran valor o que no fueran prioridad para el prototipo final, además fueron de gran ayuda al momento de la implementación de los casos de uso ya que no se contaba con diseños para las interfaces.

4. Requerimientos

Luego de finalizada la etapa de definición de la plataforma y alcance de este MVP, se procedió al refinamiento de los requerimientos funcionales y no funcionales de la plataforma.

En este capítulo se presentarán los casos de uso tanto para la plataforma web que utilizarán los usuarios finales como para la plataforma de administración que será utilizada por el equipo de operaciones de la organización.

Los casos de uso son una descripción de las actividades que pueden realizar los usuarios del sistema mediante el uso de la aplicación, buscando definir en líneas generales el comportamiento del sistema para validar los posibles escenarios y comportamientos que se espera tener. En otras palabras, describen cómo el sistema debe comportarse desde el punto de vista del usuario.

Cabe mencionar, que en los casos de uso que se describen a continuación, por **usuario** nos referimos tanto a **“Referentes”** como a **“Referidos”**. Además, los casos de uso fueron divididos por flujos dentro de la aplicación para los usuarios finales y la aplicación de administración que utilizará el directorio de la red.

Los casos de uso que se describen a continuación, son el resultado final de varias instancias de definición y refinamiento en conjunto con el equipo de REDDER.

4.1. Aplicación web (para usuarios finales)

A continuación se presentan algunos de los casos de uso más relevantes de la aplicación web para los usuarios finales (“Redderers” y “Referentes”).

4.1.1. Referidos al manejo de sesión

En esta sección se presentan los casos de uso relativos al manejo de sesión: *Registrar Referido*, *Registrar Referente*, *Iniciar Sesión*, *Cerrar Sesión*, *Cambiar contraseña* y *Recuperar contraseña*. Estos se muestran en las Figuras 4, 5, 6, 7, 8 y 9 respectivamente.

Registrar Referido
<p>Descripción: Como Referido quiero registrarme en la plataforma para obtener ayuda de un Referente en mi área.</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - Registro normal: <ul style="list-style-type: none"> - El Referido deberá ingresar: <ul style="list-style-type: none"> - Nombre de usuario - Correo - Contraseña - Sexo - Fecha de nacimiento - ... - Registro con Google/Linkedin <ul style="list-style-type: none"> - Al seleccionar la opción de "Login con Google/Linkedin" la plataforma utilizará la API de Google/Linkedin para realizar la autenticación. Luego de esto el Referente deberá ingresar los datos requeridos por la plataforma.
<p>Notas:</p> <ul style="list-style-type: none"> - El Referido no debe estar registrado. - El correo del Referido no puede existir en el sistema (debe ser único por usuario) - Un Referido no podrá tener además el rol de Referente

Figura 4: *Historia de usuario para Registrar Referido*

Registrar Referente
<p>Descripción: Como Referente quiero registrarme en la plataforma para poder mentorear a otros usuarios.</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - Registro normal <ul style="list-style-type: none"> - El Referente deberá ingresar: <ul style="list-style-type: none"> - Nombre de usuario - Correo - Contraseña - Areas de experiencia - Descripción del usuario - ... - Registro con Google/Linkedin <ul style="list-style-type: none"> - Al seleccionar la opción de "Login con Google/Linkedin" la plataforma utilizará la API de Google/Linkedin para realizar la autenticación. Luego de esto el Referente deberá ingresar los datos requeridos por la plataforma. - Luego del registro, queda una solicitud de registro de referente en la plataforma de Backoffice.
<p>Notas:</p> <ul style="list-style-type: none"> - El Referente no debe estar registrado. - El correo del Referente no puede existir en el sistema (debe ser único por usuario) - El Referente no podrá tener además el rol de Referido

Figura 5: *Historia de usuario para Registrar Referente*

Iniciar Sesión
<p>Descripción: Como Usuario quiero iniciar sesión en la plataforma para poder hacer uso de la misma.</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - Login normal: <ul style="list-style-type: none"> - El usuario deberá ingresar: <ul style="list-style-type: none"> - Nombre de usuario ó correo - Contraseña - Login Google/LinkedIn: <ul style="list-style-type: none"> - Al seleccionar la opción de "Login con Google/LinkedIn" el usuario deberá ingresar sus credenciales de Google/LinkedIn. - En caso de no estar registrado en el sistema, se procederá con el flujo de "Registro con Google/LinkedIn" mencionado en las historias de usuario "Registrar Referido" y "Registrar Referente"
<p>Notas:</p> <ul style="list-style-type: none"> - El usuario no debe estar logueado en la aplicación.

Figura 6: *Historia de usuario para Inicio de Sesión*

Como se puede observar en las historias de usuario presentadas, la autenticación mediante Google y LinkedIn fue un requerimiento de la organización para que sus usuarios pudieran autenticarse de forma más sencilla con el fin de aumentar la cantidad de usuarios en la red. Esto se detalla en profundidad en la sección [6.3.1](#)

Cerrar Sesión
<p>Descripción: Como usuario quiero cerrar sesión para proteger mis datos</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El usuario debe poder cerrar su sesión cuando considere necesario
<p>Notas:</p> <ul style="list-style-type: none"> - El usuario debe estar logueado para poder cerrar sesión

Figura 7: *Historia de usuario para Cerrar Sesión*

Cambiar Contraseña
Descripción: Como usuario quiero cambiar mi contraseña para proteger mi cuenta
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe ingresar su contraseña actual - El usuario debe ingresar su nueva contraseña

Figura 8: *Historia de usuario para Cambiar Contraseña*

Recuperar Contraseña
Descripción: Como usuario quiero recuperar mi contraseña para proteger mi cuenta
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe ingresar su nombre de usuario - El usuario recibirá un mail con los pasos a seguir - El usuario debe ingresar su nueva contraseña

Figura 9: *Historia de usuario para Recuperar Contraseña*

4.1.2. Referidos al perfil de los usuarios

En esta sección se presentan los casos de uso correspondientes al manejo del perfil de los usuarios: *Ver perfil*, *Modificar idioma de la plataforma*, *Editar Perfil* y *Editar foto de Perfil*. Estos se muestran en las Figuras 10, 11, 12 y 13 respectivamente.

Ver Perfil
Descripción: Como usuario quiero visualizar tanto mi perfil como el de otros usuarios para
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder ver su perfil o el de otros usuarios
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado

Figura 10: *Historia de usuario para Ver Perfil*

Cambiar Idioma
Descripción: Como usuario quiero cambiar el idioma en el que se muestra la plataforma
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder cambiar el idioma de la plataforma de español a inglés o viceversa

Figura 11: *Historia de usuario para cambiar el idioma de la plataforma*

Editar Perfil
Descripción: Como usuario quiero editar mi perfil para poder actualizar mis datos
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder actualizar los siguientes datos: <ul style="list-style-type: none"> - Nombre y Apellido - Pais de residencia - Teléfono - Resumen personal - Link a perfil de LinkedIn
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado

Figura 12: *Historia de usuario para Editar Perfil*

Actualizar Imagen de Perfil
Descripción: Como usuario quiero editar mi foto de perfil
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder actualizar su foto de perfil
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado - El usuario no puede estar logueado con Google o LinkedIn, en ese caso la imagen viene de la plataforma

Figura 13: *Historia de usuario para Editar foto de Perfil*

4.1.3. Referidos al flujo de selección de Referente

En esta sección se presentan los casos de uso relacionados a la selección de referentes. Estos forman parte de los flujos principales de la aplicación ya que el objetivo de la misma es la conformación de duplas de trabajo “Referente-Referido”.

Estos son *Listar Referentes*, *Marcar Referente como Favorito*, *Solicitar Referentes*, *Ver solicitudes pendientes* y *Ver el motivo de rechazo del Referente*, los cuales se presentan en las figuras 14, 15, 16, 17 y 18 respectivamente.

Listar Referentes
Descripción: Como Referido quiero visualizar la lista de los referentes que más se adecúen a mi perfil. Los perfiles a mostrar estarán pre-filtrados por mis areas de interes e idioma en el que habla
Criterios de aceptación: <ul style="list-style-type: none">- El Referido debe poder listar todas los referentes y visualizar su perfil- El Referido debe poder filtrar sobre la lista
Notas: <ul style="list-style-type: none">- El Referido debe estar logueado para poder listar los referentes- El Referido no debe estar en pareja para poder listar los referentes

Figura 14: Historia de usuario Listar Referentes

Marcar Referente como Favorito
Descripción: Como Referido quiero poder marcar a un referente como favorito para luego poder generar una solicitud.
Criterios de aceptación: <ul style="list-style-type: none">- El usuario Referido debe poder marcar cualquier referente como favorito
Notas: <ul style="list-style-type: none">- El usuario debe estar logueado para poder realizar la acción- El Referido no debe estar en pareja para poder listar los referentes

Figura 15: Historia de usuario para Marcar Favoritos

Solicitar Referentes	
Descripción:	Como Referido quiero poder generar una solicitud con como máximo tres Referente. Esas solicitudes serán procesadas luego por las referentes
Criterios de aceptación:	<ul style="list-style-type: none"> - El Referido debe poder generar una solicitud con como máximo 3 referentes y con como mínimo un referente.
Notas:	<ul style="list-style-type: none"> - El usuario debe estar logueado para poder realizar la acción - Los Referentes debe haber sido marcados como favoritos previamente - Tanto el Referido como el Referente deben estar sin pareja al momento de la solicitud

Figura 16: *Historia de usuario para Solicitar Referentes*

El requerimiento para que los usuarios “Redderers” no puedan enviar solicitudes a más de tres “Referentes” de forma simultanea fue un pedido explícito del equipo de la red pero es algo que se puede modificar en un futuro si así se desea.

Ver solicitudes pendientes	
Descripción:	Como Referido quiero poder visualizar las solicitudes que están pendientes de aprobación por los referentes.
Criterios de aceptación:	<ul style="list-style-type: none"> - El Referido debe poder visualizar las solicitud que aún se encuentran pendientes de aceptación
Notas:	<ul style="list-style-type: none"> - El usuario debe estar logueado para poder realizar la acción - El referido debe haber generado solicitudes previamente

Figura 17: *Historia de usuario para Ver Solicitudes*

Ver Motivo de Rechazo Referente
Descripción: Como Referido quiero poder visualizar el motivo por el cual mi solicitud fue rechazada por el referente mediante email.
Criterios de aceptación: <ul style="list-style-type: none"> - El Referido debe poder visualizar la razón (en caso de existir), por la que el referente rechazó su solicitud (se enviará un email)
Notas: <ul style="list-style-type: none"> - El referido debe haber generado una solicitud al referente previamente - El referente tiene que haber rechazado al referido en cuestion

Figura 18: *Historia de usuario para Ver Motivo de Rechazo*

4.1.4. Referidos al flujo de aceptación de Referidos

A continuación se presentan los casos de uso con el fin de que los “Referentes” puedan analizar y responder a las solicitudes enviadas por parte de las “Redderers”: *Listar solicitudes de Referidos a Referentes*, *Aceptar solicitud* y *Rechazar solicitud*. Estos se presentan en las figuras 19, 20 y 21 respectivamente.

Listar solicitudes de Referidos a Referentes
Descripción: Como Referente quiero ver las solicitudes que me envían los referidos
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario Referente debe poder ver todas las solicitudes que le envían los referidos
Notas: <ul style="list-style-type: none"> - El Referido debe estar logueado para poder realizar la acción - Tanto Referido como Referente no deben estar en pareja al momento

Figura 19: *Historia de usuario para Listar Solicitudes*

Aceptar solicitud
Descripción: Como Referente quiero aceptar una solicitud enviada por un referido
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario Referente debe poder aceptar una solicitud enviada por un referido
Notas: <ul style="list-style-type: none"> - El Referente debe estar logueado para poder realizar la acción - El Referente no debe tener un referido asignado al momento de Aceptar una solicitud

Figura 20: *Historia de usuario para Aceptar Solicitud*

Rechazar solicitud
Descripción: Como Referente quiero rechazar una solicitud enviada por un referido
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario Referente debe poder rechazar una solicitud enviada por un referido - El usuario Referente debe especificar de forma obligatoria las razones del rechazo - El usuario Referido recibirá la información de este rechazo via correo electrónico junto con el motivo del rechazo
Notas: <ul style="list-style-type: none"> - El Referente debe estar logueado para poder realizar la acción - El Referente no debe estar en pareja al momento de la acción

Figura 21: *Historia de usuario para Rechazar Solicitud*

4.1.5. Referidos al seguimiento de objetivos y feedback

Luego de que una “Referente” aprueba la solicitud enviada por parte de una “Referida” queda conformada una nueva dupla de trabajo o mentoría.

En esta sección se presentan los casos de uso referidos al seguimiento de la pareja: *Crear/Editar objetivo*, *Crear/Editar reunión*, *Crear/Editar nota* y *Ver objetivos y progreso*. Estos se presentan en las figuras [22](#), [23](#), [24](#) y [25](#) respectivamente.

Crear/editar objetivo
<p>Descripción: Como Usuario quiero poder crear objetivos para la pareja. Se pedirá el nombre del objetivo, así como avances por mes.</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El usuario debe poder crear hasta tres objetivos.
<p>Notas:</p> <ul style="list-style-type: none"> - El Usuario debe estar logueado para poder realizar la acción - El Usuario debe estar en pareja

Figura 22: *Historia de usuario para Crear/Editar Objetivo*

Es importante destacar que el máximo de tres objetivos fue un requerimiento impuesto por el equipo de la red pero es posible modificarlo en un futuro a la cantidad que se requiera.

Crear/editar reunión
<p>Descripción: Como Usuario quiero poder dar seguimiento de las reuniones llevadas a cabo, creando y editando las reuniones indicando el día que la reunión se llevó a cabo así como un resumen de la misma</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El usuario debe poder crear reuniones.
<p>Notas:</p> <ul style="list-style-type: none"> - El Usuario debe estar logueado para poder realizar la acción - El Usuario debe estar en pareja

Figura 23: *Historia de usuario para Crear/Editar Reunión*

Crear/editar nota
<p>Descripción: Como Usuario quiero poder dar seguimiento a la pareja pudiendo agregar notas que crea necesarias. Se podrá agregar una nota.</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El usuario debe poder crear notas.
<p>Notas:</p> <ul style="list-style-type: none"> - El Usuario debe estar logueado para poder realizar la acción - El Usuario debe estar en pareja

Figura 24: *Historia de usuario para Crear/Editar Nota*

Ver objetivos y progreso
<p>Descripción: Como Usuario quiero ver las reuniones, objetivos y notas que, en su conjunto, indican el progreso de mi pareja.</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El usuario debe poder ver los objetivos de la pareja, su respectivo progreso y el feedback en caso de haberlo enviado - El usuario debe poder ver el seguimiento de las reuniones, viendo el día y el resumen de las mismas - EL usuario debe poder ver las notas agregadas a la pareja
<p>Notas:</p> <ul style="list-style-type: none"> - El usuario debe estar logueado para poder realizar la acción - El usuario debe estar en pareja

Figura 25: *Historia de usuario para Ver Objetivo*

4.1.6. Referidos a finalización de una pareja

Por lo general una dupla o pareja de trabajo tiene una duración máxima de tres meses, esto fue un requerimiento impuesto por el equipo de la Red, detallado en la sección 6.2. En caso de que la dupla no esté siendo fructífera, o haya sucedido algún tipo de inconveniente, ambos usuarios tendrán la posibilidad de finalizar la pareja mediante los siguientes casos de uso: *Cancelar pareja* (Figura 26), *Denunciar pareja* (Figura 27). Además,

siempre que una pareja finaliza (ya sea por haber transcurrido tres meses o por cancelación), se puede puntuar a la referente y así tener más información sobre los mismos. Este caso de uso es llamado *Puntuar Referente* (Figura 28).

Cancelar pareja
Descripción: Como usuario quiero cancelar la relación de pareja con mi Referente/Referido.
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder cancelar el emparejamiento, siempre brindando un motivo por el cual se está cancelando - La pareja queda automáticamente cancelada
Notas: <ul style="list-style-type: none"> - El Usuario debe estar logueado para poder realizar la acción - El Usuario debe estar en una pareja - El Usuario brinda una descripción del porqué de la cancelación de la pareja

Figura 26: *Historia de usuario para Cancelar Pareja*

Denunciar pareja
Descripción: Como Usuario quiero poder denunciar a una Referente/Referida.
Criterios de aceptación: <ul style="list-style-type: none"> - El Usuario debe poder denunciar a su pareja, siempre brindando el motivo de la denuncia y una descripción de la misma. - La Pareja queda automáticamente cancelada
Notas: <ul style="list-style-type: none"> - El Usuario debe estar logueado para poder realizar la acción - El Usuario debe estar en una pareja

Figura 27: *Historia de usuario para Denunciar Pareja*

Puntuar referente
<p>Descripción: Una vez que la pareja culmina, como Referido quiero poder puntuar la actuación de mi Referente a lo largo de la pareja.</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El Referido debe poder puntuar la actuación de la pareja al finalizar el emparejamiento.
<p>Notas:</p> <ul style="list-style-type: none"> - El Referido debe estar logueado para poder realizar la acción - La pareja debe haber finalizado para realizar la puntuación de la misma - Esta información no será pública en ningún momento, será únicamente visible desde el Backoffice

Figura 28: *Historia de usuario para Puntuar Referente*

4.2. Aplicación de administración

En esta sección se presentan los casos de uso para la plataforma de administración o “backoffice”. Como ya se mencionó esta aplicación tiene como objetivo facilitar al equipo de la gestión y operaciones de la red.

A diferencia de los casos de uso presentados para la aplicación web que usarán los usuarios finales (“Redderers“ y “Referentes”), cuando hablamos de *Usuario* en este contexto se hace referencia a los *Usuarios de Administración o Backoffice*. En un principio es el equipo del directorio de REDDER quien utilizará esta aplicación pero será posible agregar nuevos usuarios en caso que sea necesario.

Al igual que en la sección anterior se presentan algunos de los casos de uso divididos por flujos dentro de la aplicación de administración.

4.2.1. Referidos al manejo de sesión

En esta sección se presentan los casos de uso relativos al manejo de sesión de la plataforma de administración: *Iniciar Sesión*, *Cerrar Sesión*, *Cambiar contraseña*, *Recuperar contraseña* y *Modificar el idioma de la plataforma*. Estos se presentan en las figuras 29, 30, 31, 32 y 33 respectivamente.

Iniciar Sesión
Descripción: Como Usuario quiero iniciar sesión en la plataforma para poder hacer uso de la misma.
Criterios de aceptación: El usuario deberá ingresar: <ul style="list-style-type: none"> - Nombre de usuario ó correo - Contraseña
Notas: El usuario no debe estar autenticado en la aplicación.

Figura 29: *Historia de usuario para Iniciar Sesión*

Cerrar Sesión
Descripción: Como usuario quiero cerrar sesión
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder cerrar su sesión cuando considere necesario
Notas: <ul style="list-style-type: none"> - El usuario debe estar autenticado para poder cerrar sesión

Figura 30: *Historia de usuario para Cerrar Sesión*

Cambiar Contraseña
Descripción: Como usuario quiero poder cambiar mi contraseña.
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe ingresar su contraseña actual - El usuario debe ingresar su nueva contraseña
Notas: <ul style="list-style-type: none"> - El usuario debe estar autenticado para esta acción

Figura 31: *Historia de usuario para Cambiar Contraseña*

Recuperar Contraseña
Descripción: Como usuario quiero recuperar mi contraseña
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe ingresar su nombre de usuario o email - El usuario recibirá un mail con los pasos a seguir - El usuario debe ingresar su nueva contraseña
Notas: <ul style="list-style-type: none"> - El usuario no debe estar autenticado para esta acción

Figura 32: *Historia de usuario para Recuperar Contraseña*

Cambiar Idioma
Descripción: Como usuario quiero cambiar el idioma en el que se muestra la plataforma
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder cambiar el idioma de la plataforma de español a inglés o viceversa

Figura 33: *Historia de usuario para cambiar el idioma de la plataforma*

4.2.2. Referidos al manejo de usuarios de administración

En esta sección se presentan los casos de uso relativos al manejo de usuarios de la plataforma de administración: *Agregar Usuario* (Figura 34), *Eliminar Usuario* (Figura 35) y *Listar Usuarios* (Figura 36).

Agregar Usuario Backoffice
Descripción: Como usuario quiero poder crear un nuevo usuario de backoffice
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder ingresar los datos del nuevo usuario backoffice
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción - El usuario debe ser Administrador de Backoffice

Figura 34: *Historia de usuario para Agregar Usuario*

Eliminar Usuario Backoffice
Descripción: Como usuario quiero poder eliminar un usuario de backoffice para que no pueda volver a acceder a la plataforma
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder eliminar un usuario de backoffice - Luego de ser eliminado, el usuario eliminado no podrá ingresar a la plataforma
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción - El usuario debe ser Administrador de Backoffice

Figura 35: *Historia de usuario para Eliminar Usuario*

Listar Usuarios Backoffice
Descripción: Como usuario quiero poder ver todos los usuarios de Backoffice
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder listar todos los usuarios de backoffice - Se podrá filtrar por nombre
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción - El usuario debe ser Administrador de Backoffice

Figura 36: *Historia de usuario para Listar Usuarios*

4.2.3. Referidos al manejo de la aplicación

A continuación se presentan los casos de uso relativos a la gestión y uso de la aplicación: *Listar usuarios de la aplicación*, *Ver denuncias sobre los usuarios de la aplicación*, *Responder a denuncia*, *Bloquear usuario*, *Desbloquear usuario*, *Ver reportes de uso*, *Responder a denuncia*, *Agregar área de interés*, *Actualizar área de interés*, *Eliminar área de interés*, *Listar solicitudes de referente* y *Responder solicitudes de referente*. Estos se presentan en las figuras 37, 38, 39, 40, 41, 42, 43, 44, 45, 47 y 48 respectivamente.

Listar Usuarios de aplicación
Descripción: Como usuario quiero poder ver todos los Referentes/Referidos ingresados en el sistema
Criterios de aceptación: <ul style="list-style-type: none">- El usuario debe poder listar todos los Referentes/Referidos- Se podrá filtrar por nombre
Notas: <ul style="list-style-type: none">- El usuario debe estar logueado para realizar la acción

Figura 37: *Historia de usuario para Listar Usuarios*

Ver denuncias sobre los usuarios de la aplicación
Descripción: Como usuario quiero poder ver las denuncias sobre los usuarios de la aplicación
Criterios de aceptación: <ul style="list-style-type: none">- El usuario debe poder listar todas las denuncias realizadas a los usuarios de la aplicación
Notas: <ul style="list-style-type: none">- El usuario debe estar logueado para realizar la acción

Figura 38: *Historia de usuario para Ver Denuncias*

Responder a Denuncia
<p>Descripción: Como usuario quiero poder responder a una denuncia sobre un usuario de la aplicación</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El usuario debe poder desestimar una denuncia realizada sobre un usuario de la aplicación - El usuario debe poder confirmar una denuncia realizada sobre un usuario de la aplicación. En este caso, el usuario de la aplicación quedará bloqueado sin poder ingresar a la plataforma
<p>Notas:</p> <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción

Figura 39: *Historia de usuario para Responder a Denuncia*

Bloquear Usuario (Referido/Referente)
<p>Descripción: Como usuario quiero poder bloquear un usuario del sistema</p>
<p>Criterios de aceptación:</p> <ul style="list-style-type: none"> - El usuario debe poder bloquear un usuario del sistema - Luego de bloqueado el usuario, este usuario no se podrá volver a loguear en el sistema. - Luego de bloqueado, no se podrá crear una cuenta con ese mismo mail
<p>Notas:</p> <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción - El usuario a bloquear, debe estar previamente habilitado

Figura 40: *Historia de usuario para Bloquear Usuario*

Desbloquear Usuario (Referido/Referente)
Descripción: Como usuario quiero poder desbloquear un usuario del sistema
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder desbloquear un usuario del sistema - Luego de desbloqueado el usuario, este usuario podrá volver a utilizar el sistema
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción - El usuario a desbloquear, debe estar previamente bloqueado

Figura 41: *Historia de usuario para Desbloquear Usuario*

Agregar área de Interés
Descripción: Como usuario quiero poder agregar una nueva área de interés
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder agregar un area de interés indicado el nombre y con qué otras areas de interés 'matchea'
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción

Figura 42: *Historia de usuario para Agregar Área de Interés*

Actualizar área de interés
Descripción: Como usuario quiero poder actualizar un área de interés
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder actualizar un area de interés, ya sea el nombre y con qué otras areas de interés 'matchea'
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción

Figura 43: *Historia de usuario para Actualizar Área de Interés*

Eliminar área de Interés
Descripción: Como usuario quiero poder eliminar un área de interés
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder eliminar un area de interés
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción

Figura 44: *Historia de usuario para Eliminar Área de Interés*

Ver reportes de uso
Descripción: Como usuario quiero poder ver los reportes de la aplicación
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder ver los reportes generados por la plataforma de administración para poder sacar conclusiones de uso
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción

Figura 45: *Historia de usuario para Ver Reportes*

El equipo de la red solicitó tener acceso a ciertas métricas y reportes en la pantalla de inicio de la aplicación de gestión. Algunas de estas son la cantidad de usuarios “Referentes” y “Referidos” que hay registrados y la cantidad de parejas agrupadas por estado, donde:

- **Activas:** Son las parejas activas al momento, es decir, que siguen en la relación de mentoría
- **Evaluación Pendiente:** Son las parejas que finalizaron (ya sea por que alcanzaron el tiempo de duración máximo establecido o por la cancelación de la misma) en donde el “Referido” aún no a puntuado a la “Referente”.
- **Completadas:** Son las parejas completadas en donde el “Referido” ya puntuó a la “Referente”.

- **Canceladas:** Son las parejas canceladas por cualquiera de los dos usuarios.

Además de los reportes mencionados, se puede ver el promedio de reuniones llevadas a cabo por pareja así como también los objetivos y progreso de los mismos.

En la Figura 46 se puede ver una captura de la pantalla de inicio de la plataforma de administración donde se muestran éstos reportes.

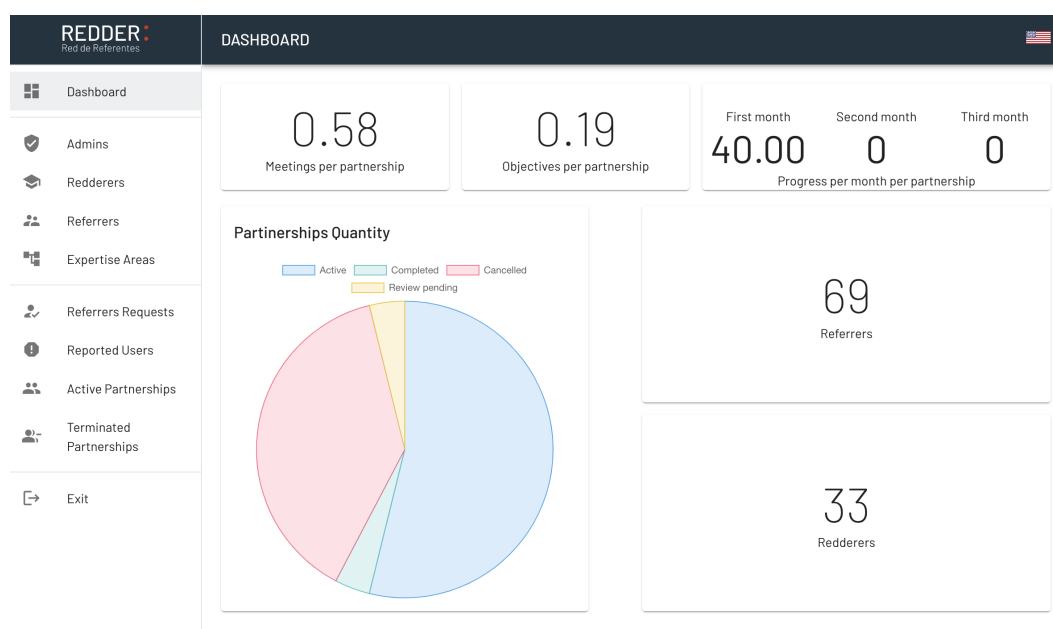


Figura 46: Pantalla de inicio de la aplicación de administración

Listar solicitudes de Referentes
Descripción: Como usuario quiero poder visualizar las solicitudes para ser referentes.
Criterios de aceptación: <ul style="list-style-type: none"> - El usuario debe poder listar todas las solicitudes - Se podrá ver toda la información de la solicitud
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción

Figura 47: *Historia de Usuario para Listar Solicitudes de Referentes*

Aceptar/Rechazar solicitud de Referente
Descripción: Como usuario quiero poder aceptar o rechazar una solicitud para ser referente
Criterios de aceptación: <ul style="list-style-type: none"> - El usuarios debe poder aceptar o rechazar cualquier solicitud - Se podrá ingresar una razón de porqué se rechazó la solicitud
Notas: <ul style="list-style-type: none"> - El usuario debe estar logueado para realizar la acción

Figura 48: *Historia de Usuario para Aceptar/Rechazar Solicitudes*

Estos dos últimos casos de uso que se presentaron y que tienen como fin poder ver y analizar las solicitudes de registro de los “Referentes” que desean unirse a la red, fue algo en que el directorio de la organización hizo énfasis ya que el poder controlar quien es aceptado y quien no como “Referente” en la plataforma ayuda a brindar un entorno serio y seguro para todos los integrantes del sistema.

A continuación se presenta un diagrama de secuencia para ilustrar este caso de uso de forma más clara:

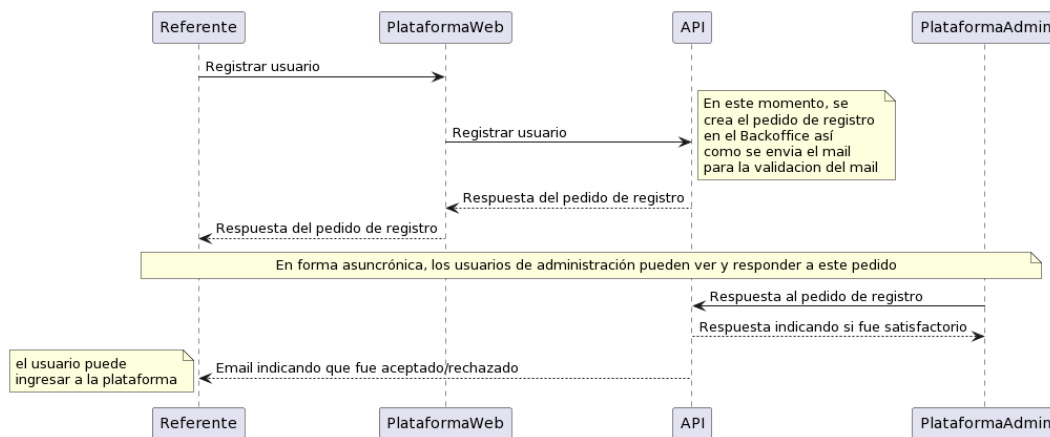


Figura 49: DSS de Registro de Referentes

Todos los requerimientos presentados en las secciones 4.1 y 4.2 son resultado de un proceso iterativo e incremental con el equipo de REDDER donde se fueron refinando los requisitos paso a paso. Todos los requerimientos presentados anteriormente son los requerimientos finales luego del proceso de refinamiento de los mismos.

4.3. Requerimientos no funcionales

A continuación se presentan los requerimientos no funcionales del sistema a implementar.

4.3.1. Concurrencia

Se entiende por concurrencia, la habilidad del sistema de poder ejecutar el programa (o partes de el) en desorden o parcialmente sin afectar el resultado final [10]. Se estimó que la plataforma llegue a unos 500 usuarios en total en el próximo año, por lo que se espera que la plataforma puede ser utilizada por al menos 500 usuarios de forma concurrente sin afectar su rendimiento.

4.3.2. Usabilidad

Siguiendo las normas ISO 25000, relativas a la calidad de software y datos [10], se entiende por usabilidad a la capacidad de un sistema de ser entendido,

aprendido, usado y resultar atractivo para el usuario. Debido a los usuarios objetivos del sistema en cuestión, no se podía asumir que todos ellos tienen facilidad para el uso de tecnologías.

Por todo lo expuesto anteriormente, el sistema deberá presentar una interfaz *amigable*, para que la aplicación pueda ser utilizada de forma sencilla por todos sus usuarios.

Este requerimiento se fue verificando iterativamente con el equipo de REDDER, mostrando los requerimientos implementados, escuchando retroalimentación en todo momento y realizando mejoras.

4.3.3. Mantenibilidad

Nuevamente, teniendo en cuenta las normas ISO 25000 pero relativas a la mantenibilidad [11], se entiende la misma como la característica y capacidad del producto de ser modificado efectiva y eficientemente, debido a diferentes necesidades. Es claro que al realizar la implementación de esta plataforma en el contexto de un proyecto de grado es probable que la implementación de las mismas la continúen otros profesionales por lo que este requerimiento es importante en este caso.

Este requerimiento no funcional nace directamente de la esencia del sistema, ya que si la misma empieza a ser utilizada por mucha gente, seguramente se querrán incorporar nuevas funcionalidades, como puede ser la integración de un chat de mensajería directa dentro de la aplicación o la posibilidad de integrar el calendario de cada usuario para coordinar reuniones así como videoconferencia, por nombrar alguna de todas las posibilidades de crecimiento que existen.

La verificación de este requerimiento se fue realizando también de forma iterativa con el proceso de verificación de código, el cual se presenta en la sección 8.1

4.3.4. Escalabilidad

Se entiende por escalabilidad, la capacidad para ofrecer un servicio de alta calidad a medida que aumentan las demandas sobre el sistema [12], donde hay dos tipos de escalabilidad, *scaling-up*, haciendo al sistema más potente, o *scaling-out*, agregando más instancias del sistema para soportar la alta

demanda. Se estimó que la plataforma llegue a unos 500 usuarios en total en el próximo año, por lo que este requerimiento es importante para que la plataforma no sufra pérdidas de rendimiento en el futuro.

Este requerimiento fue resuelto en el diseño de la plataforma, como se describe en el capítulo 5.

4.3.5. Web responsiva

Se entiende por web responsiva toda web que sea capaz de adaptarse a cualquier dispositivo donde se visualice. Tal como se puede ver en el artículo “*El 91,5 % de los internautas ya accede a Internet a través del móvil*” [13] publicado por el diario *El País*, hoy en día la mayoría de los usuarios hace uso de las plataformas web a través de sus *smartphones*. Por lo tanto, con el fin de facilitar la adopción y retención de usuarios es que la plataforma debe ser responsiva.

Este requerimiento fue verificado durante la implementación del sistema.

4.3.6. Seguridad y privacidad de los datos

Dado que la plataforma solicita y almacena datos personales de los usuarios, como nombre y apellido, país de residencia, teléfono, entre otros, es importante que la misma brinde seguridad y privacidad para los datos de los usuarios registrados.

Este requerimiento se tuvo en cuenta al realizar el diseño del sistema (capítulo 5) y al elegir la plataforma de despliegue para la solución final (capítulo 7).

5. Diseño de la solución

En este capítulo se presenta el diseño del sistema a construir. En primer lugar se argumenta acerca de la arquitectura del sistema y luego se presenta el diseño de la base de datos.

5.1. Arquitectura del sistema

Se utilizó una arquitectura Cliente/Servidor, en este tipo de arquitecturas se distinguen dos componentes bien definidos: cliente y servidor [14]. En la Figura 50 se puede ver un diagrama de esto.

Los servidores son los encargados de proveer respuestas de recursos o servicios ante las solicitudes que envían los clientes. Es interesante notar que el servidor no necesariamente se ejecuta en una sola maquina y hay varios tipos de servidores como web, de archivo, de correo, entre otros. Los clientes se consideran el componente activo en este tipo de arquitecturas ya que son los encargados de realizar las solicitudes mientras que el servidor está a la espera de estas solicitudes.

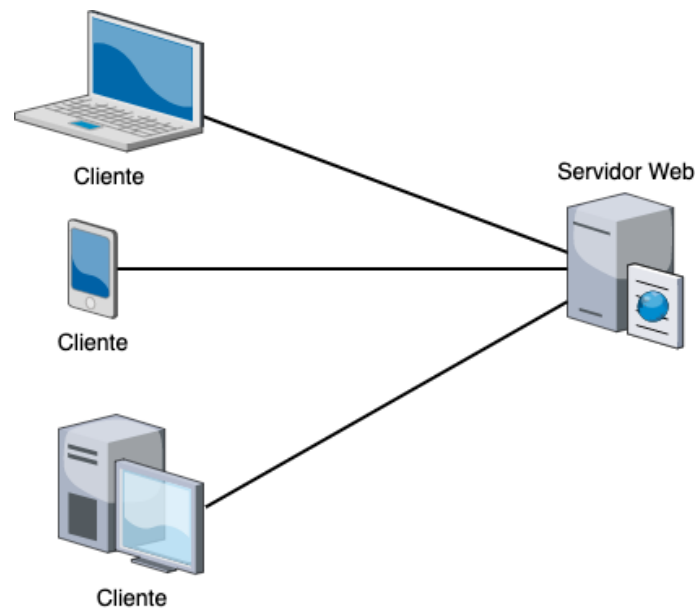


Figura 50: *Arquitectura Cliente/Servidor*

Se decidió utilizar este tipo de arquitectura por las ventajas que presenta la misma [14]. Algunas de estas ventajas son:

- Separación de responsabilidades: el cliente se encarga de realizar solicitudes y desplegar la información al usuario y el servidor de atender, procesar y dar respuesta a los clientes
- Escalabilidad: Tanto horizontal (agregando más nodos al sistema) como vertical (agregando recursos a un nodo)
- Centralización de los datos: el servidor es el único componente que tiene acceso a los datos

Según la reconocida empresa Gartner Group una arquitectura de este tipo (cliente/servidor) se puede modelar en dos o tres capas físicas. En este caso se tiene una arquitectura con tres capas físicas ya que tenemos el componente cliente, el servidor y la base de datos como se puede ver en la Figura 51. [15]

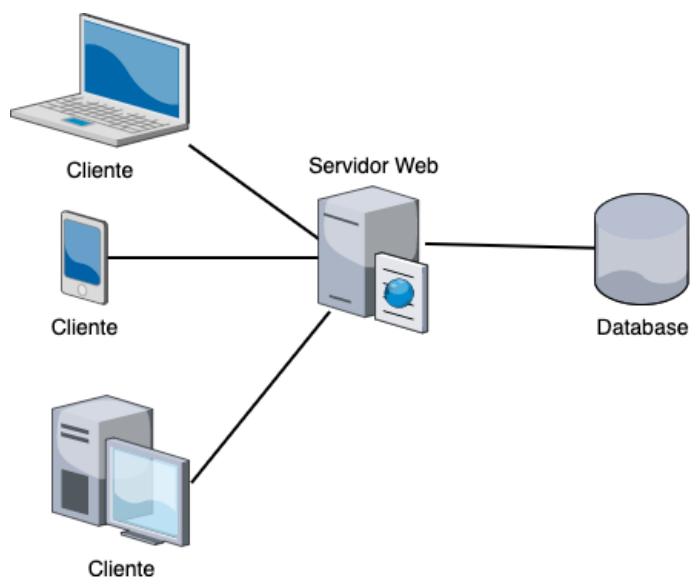


Figura 51: *Arquitectura Cliente/Servidor REDDER*

Entrando un poco más en detalle dentro del diseño del sistema, en la Figura 52 se muestra la división en componentes de las distintas capas que conforman

la arquitectura de la plataforma Redder. En la misma se puede ver que la única división en múltiples componentes se presenta dentro de la capa de cliente, en donde se dividen las responsabilidades entre dos componentes separados. El primer componente, llamado en la Figura 52 como *REDDER App* hace referencia a la aplicación web para los usuarios finales, mientras que *REDDER Backoffice* hace referencia a la plataforma de administración.

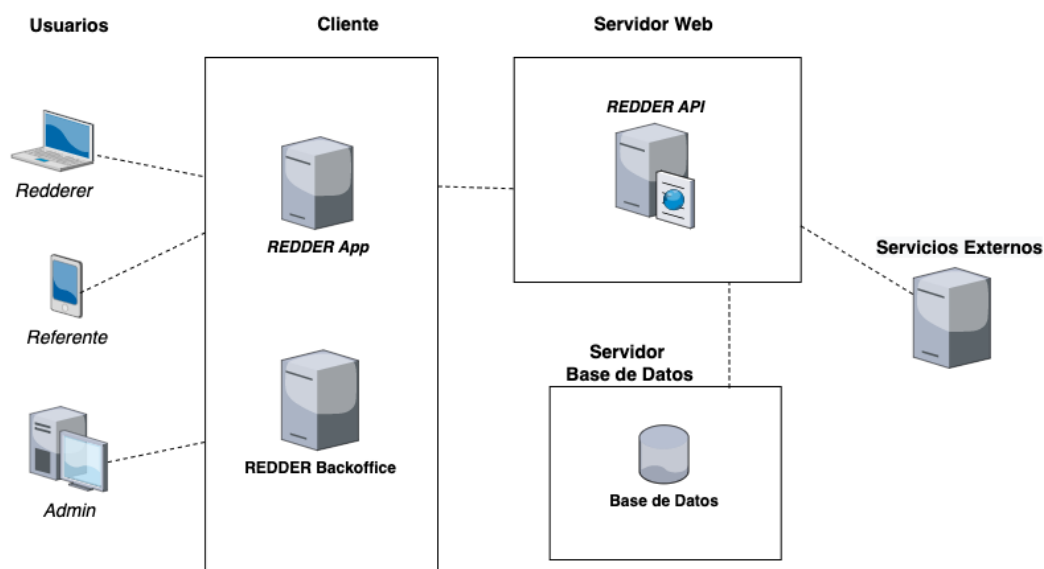


Figura 52: *Arquitectura Plataforma Redder*

5.2. Diseño base de datos

En la Figura 53 se puede observar un diagrama conocido como MER (Modelo Entidad-Relación) en el cual se muestra como se representan las entidades del sistema en la base de datos.

A continuación se mencionan cada una de las entidades con una breve descripción:

- **BOUser:** Son los usuarios que tienen acceso a la plataforma de administración o Backoffice. Desde el Backoffice se pueden crear y eliminar otros usuarios.
- **User:** Son los usuarios finales de la plataforma.

- **ExpertiseAreas:** Son las áreas de interés utilizadas por los usuarios finales para indicar sus preferencias.

Es utilizando estas áreas que se sugieren ciertos “Referentes” a los “Referidos”.

Se pueden crear, editar o eliminar desde la plataforma de administración.

- **Report:** Los reportes son utilizados para almacenar las denuncias entre usuarios de la plataforma.

Se despliegan en el backoffice con la razón por la cual se reportó al usuario.

- **Favorite:** Esta entidad es utilizada por parte de los “Redderers” para almacenar los perfiles de los “Referentes” con los cuales les interesa conformar una pareja.

- **Partnership:** Es la entidad central de la aplicación. En esta colección, se mantiene toda la información relacionada a las parejas.

- **Cron Jobs:** Son entidades utilizadas para el manejo de las tareas programadas.

Algunas de las tareas programadas que se utilizan en el contexto de este sistema tienen como fin terminar las parejas luego de tres meses de iniciadas.

Se mantienen y utilizan a través de una librería llamada Agenda [16].

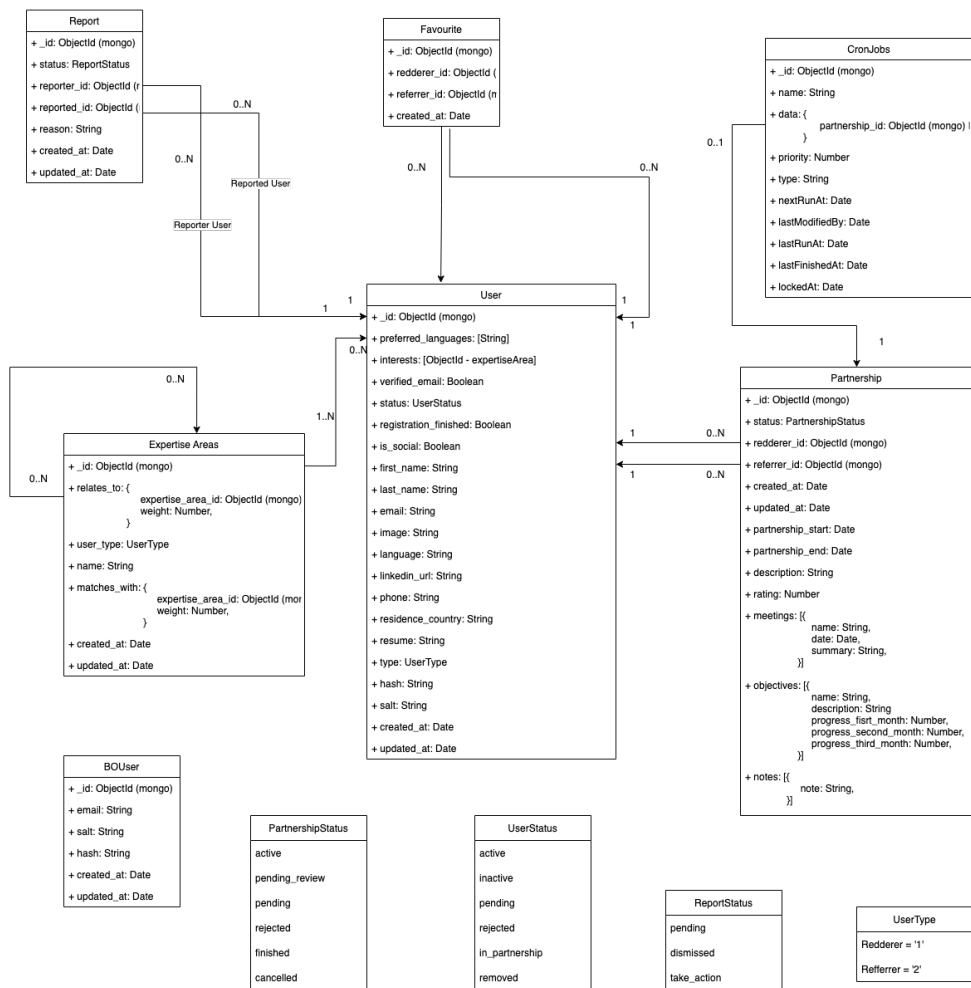


Figura 53: Diagrama de la base de datos

En el anexo F se pueden observar con mayor detalle cada una de las entidades con sus respectivos atributos.

6. Implementación de la solución

En este capítulo se profundiza acerca de las tecnologías y herramientas utilizadas para implementar la arquitectura planteada en la capítulo 5.

6.1. Descripción general

En esta sección se describe la implementación de la arquitectura presentada en la sección 5.1 y en las subsecciones se describe cada uno de los subsistemas desarrollados.

En la figura 54 se muestra la arquitectura utilizada de Cliente/Servidor como se presentó en el capítulo anterior utilizando el stack MERN (MongoDB, ExpressJS, ReactJS y NodeJS) ya que se ajustaba a los requerimientos y arquitectura del sistema además de que el equipo contaba con experiencia en estas tecnologías, siendo éste un stack muy utilizado con una comunidad muy grande lo cual facilita el acceso a información. [17]

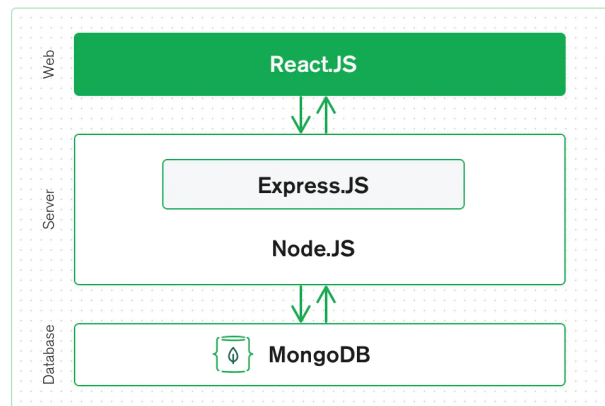


Figura 54: *MERN Stack*

En la Figura 55 se pueden observar los tres componentes de la arquitectura presentados en la sección anterior y dentro de cada uno de ellos las tecnologías que se utilizaron para su implementación. Se decidió también

agregar Typescript [18], tanto en las aplicaciones web como en el servidor (API) para obtener código tipado con el fin de minimizar errores y obtener código más mantenible.

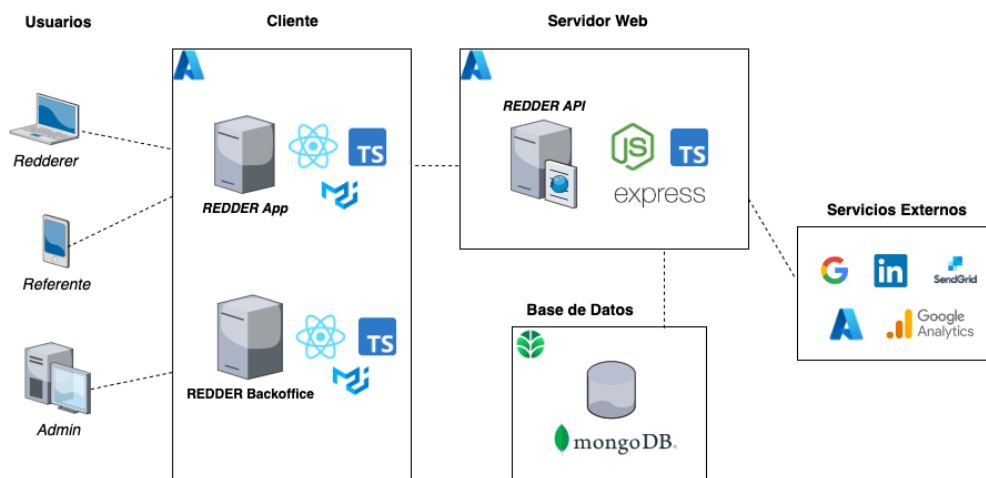


Figura 55: Arquitectura Plataforma Redder indicando las tecnologías utilizadas

Teniendo en cuenta las tecnologías utilizadas, en la Figura 55 se puede visualizar la arquitectura completa de Redder indicando la tecnología utilizada para la implementación de cada uno de los componentes. Como se puede observar en ella, el sistema está compuesto de los siguientes componentes:

- Aplicación web para usuarios finales
- Aplicación web de Backoffice o Administración
- Servidor o API
- Base de Datos
- Servicios externos

A continuación se describe la implementación y función de cada uno de los componentes del sistema, las integraciones con los servicios externos como Sendgrid para el envío de correos y Google o LinkedIn para la autenticación de usuarios se encuentra en la sección 6.3.

Con respecto a la plataforma cliente, es la utilizada por los usuarios finales. Esto quiere decir que tanto las “*Redderers*” como las “*Referentes*” la utilizarán con el fin de conformar una dupla de trabajo, fijar objetivos y poder realizar un seguimiento del progreso de los mismos.

6.1.1. Aplicaciones web para usuarios finales

Como ya se mencionó esta es la aplicación principal del sistema ya que será utilizada por los usuarios finales (“Redderers” y “Referentes”).

Esta aplicación web se implementó utilizando ReactJS [19] como tecnología principal debido a que es una de las librerías de código abierto más utilizadas hoy en día para el desarrollo de interfaces de usuario, cuenta con una gran comunidad y el equipo de desarrollo contaba con experiencia utilizándola.

Para la implementación de los componentes de la interfaz se utilizó Material-UI, una librería muy utilizada para el desarrollo de aplicaciones web, con su principal ventaja siendo la posibilidad de reutilizar componentes ya diseñados e implementados. Esta herramienta nos ayudó mucho en la etapa de implementación ya que no se contaba con un diseñador en el equipo.

También se utilizó Redux [20] para el manejo del estado y datos de la aplicaciones. Para la comunicación con la API se utiliza la librería Axios [21], enviando las peticiones por medio de HTTPS para una conexión segura.

Otra de las ventajas de utilizar React para la implementación de la plataforma es la posibilidad de crear y reutilizar componentes en la plataforma ayudando a la mantenibilidad de la misma ya que con este enfoque se reduce de forma significativa la cantidad de código.

La plataforma web se puede subdividir en capas tal como se puede ver en la Figura 56. En la capa de presentación están presentes los componentes React implementados con Material-UI y en la capa de servicios esta la lógica utilizada para la comunicación vía HTTPS con la API mediante la librería Axios.

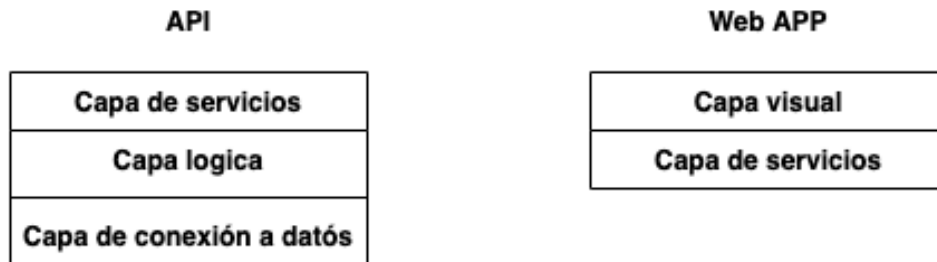


Figura 56: *Diagrama de capas*

6.1.2. Backoffice o Aplicación de administración

Como ya se mencionó anteriormente en la sección 4.2.3, la plataforma de administración es una aplicación web que tiene como fin facilitar la gestión de la plataforma, en principio será utilizada por el equipo de REDDER para monitorear y gestionar la red de una manera más eficiente.

Algunas de las funcionalidades principales de esta aplicación es el control del registro de los Referente, el despliegue de métricas sobre los usuarios y parejas, el manejo de las áreas de interés y de los usuarios permitiendo la aceptación y bloqueo de los mismos con el fin de que REDDER sea una plataforma segura.

En lo que refiere a la implementación, se utilizaron las mismas tecnologías que en la aplicación web para los usuarios finales descrita en la sección anterior.

6.1.3. Base de datos

Para la implementación de la base de datos se decidió utilizar MongoDB, una base de datos documental. Esta decisión se debió en gran parte a que los modelos de datos de las entidades que conforman el sistema fueron variando desde el comienzo del proyecto.

El uso de base de datos documentales permitió un gran ahorro de tiempo tanto en la configuración inicial de la misma como a la hora de crear y correr las migraciones necesarias para actualizar los modelos y datos en comparación con una base de datos relacional.

Además, la experiencia previa de los integrantes del equipo de desarrollo en otros proyectos utilizando base de datos documentales influyó a la hora de tomar esta decisión.

6.1.4. API o Backend

El backend está construido como una API REST utilizando Node.js con el framework Express.js [22] y el uso de TypeScript [18]. Se decidió usar Node con Express debido a la experiencia previa del equipo en estas tecnologías, además de que las mismas tienen una buena documentación y una comunidad que las respalda, algo importante al momento de hablar sobre mantenibilidad. Además la elección de añadir TypeScript y construir este componente en capas, apunta claramente a satisfacer este mismo requerimiento, la mantenibilidad; uno de los requisitos no funcionales más importantes.

Este componente está encargado de encapsular toda la lógica y las reglas de negocio del sistema, además de ser el encargado de la comunicación con la base de datos, pudiendo de esta manera, persistir los datos necesarios.

En la Figura 56 se observan las capas que presenta y a continuación se describen las mismas:

La capa de servicios contiene los endpoints que van a ser consumidos por ambas aplicaciones de frontend. Presenta las rutas y se encarga de la validación de los datos de entrada. En la capa lógica, se encuentra toda la lógica de negocio del sistema; la capa de conexión a datos es la encargada de la conexión a la base de datos MongoDB utilizando Mongoose como ORM.

Como ya se mencionó, al implementar la API se trató de que seguir los lineamientos que plantea REST [23] para que esta sea autoexplicativa, siendo de esta manera más mantenible por futuros desarrolladores. En el Anexo D se adjunta una imagen de la colección de Postman con los endpoints que expone esta API.

Se utilizaron diferentes librerías para las diferentes funcionalidades de la aplicación, como puede ser Agenda [16], ya expuesto anteriormente, o Passport.js [24], utilizada para el manejo de la autenticación en la plataforma. Ésta maneja la misma de forma segura a través de un JSON

Web Token (JWT), agregándole encriptado para las contraseñas en la base de datos.

La comunicación de la aplicación se maneja mediante correos electrónicos; para esto se encuentra integrada a *Sendgrid* [25], una plataforma frecuentemente utilizada para este fin por su simplicidad y bajo costo. Más adelante se detallará la misma con más profundidad.

6.2. Tareas programadas

Como se mencionó anteriormente, las tareas programadas son una entidad central en esta arquitectura.

Para la implementación de este módulo se realizó un estudio y pruebas de varias librerías resultando esta la más intuitiva y útil para este contexto. Esto permitió crear una solución a los requerimientos de forma eficiente sin la necesidad de ejecutar una tarea de forma diaria optimizando así los recursos necesarios para la aplicación. Ejemplos de éste último caso, pueden ser las librerías de *node-schedule* [26] o *node-cron* [27], que quedaron descartadas por lo antes expuesto. Por otro lado, encontramos las librerías de *Bottleneck* [28], *Bull* [29] y *Agenda* [16] como librerías que ayudan al manejo de tareas programadas sin la necesidad de ejecutar un *cron* diariamente. Finalmente se eligió *Agenda* ya que era la única que soportaba a la base de datos que se utiliza (MongoDB) para guardar las tareas programadas, al contrario de las otras dos mencionadas que utilizan Redis [30], un motor de base de datos en memoria.

Las tareas programadas son mantenidas y ejecutadas a través de la librería *Agenda*, como ya se menciona. En el momento en que una pareja es dada de alta, es decir, la “Referente” acepta una petición previamente hecha por un usuario “Redderer”, se crea una tarea programada para dentro de 3 meses exactamente. Esta tarea es la encargada de ejecutar una función que da de baja las parejas que superan los tres meses de duración y envía correos a modo de información. Esta funcionalidad esta contenida dentro de un mismo modulo, y separado del resto, por lo que es fácilmente adaptable o extensible si es requerido en el futuro.

6.3. Integraciones

Hoy en día es muy común que cualquier plataforma web tenga la necesidad de integrarse con diversos servicios externos para brindar la mejor experiencia posible a sus usuarios y la plataforma de REDDER no es la excepción.

A continuación se mencionan cada una de las integraciones y la utilidad de cada una de ellas.

6.3.1. Autenticación de Google y LinkedIn

Esta es una integración muy común en aplicaciones web modernas, brindándole al usuario la posibilidad de utilizar sus redes sociales para autenticarse en diferentes sistemas. En el caso de REDDER se implementó la integración para autenticación con Google y LinkedIn.

La plataforma fue integradas de manera similar, utilizando la librería Passport [24]. Para la implementación de estas integraciones fueron necesarias algunas configuraciones en ambos sistemas (app y backoffice).

A alto nivel la autenticación funciona de la siguiente manera. Primero se hace una petición inicial redireccionando al usuario a la plataforma con la cual desea ingresar al sistema. En ella, se le pide autenticarse (si no lo estaba previamente) y luego la plataforma (Google o LinkedIn) redireccionan a una ruta de la plataforma de REDDER previamente configurada con la información del usuario que está ingresando. En la integración en cuestión pedimos únicamente la información básica, esto es, imagen, nombre, apellido y el correo electrónico. En la plataforma se tiene la posibilidad de pedir más información si así se desea, siempre y cuando los usuarios acepten otorgarla. Luego de obtenida esta información, se genera un usuario con falta de datos necesarios para utilizar la plataforma (por ejemplo los intereses del mismo). Por lo tanto, se requiere que el mismo termine de llenar esa información. Una vez terminado este paso, se crea un JSON Web Token, utilizado para la identificación del usuario. En la Figura 57 se puede ver un diagrama de como funciona el flujo de autenticación previamente descrito.

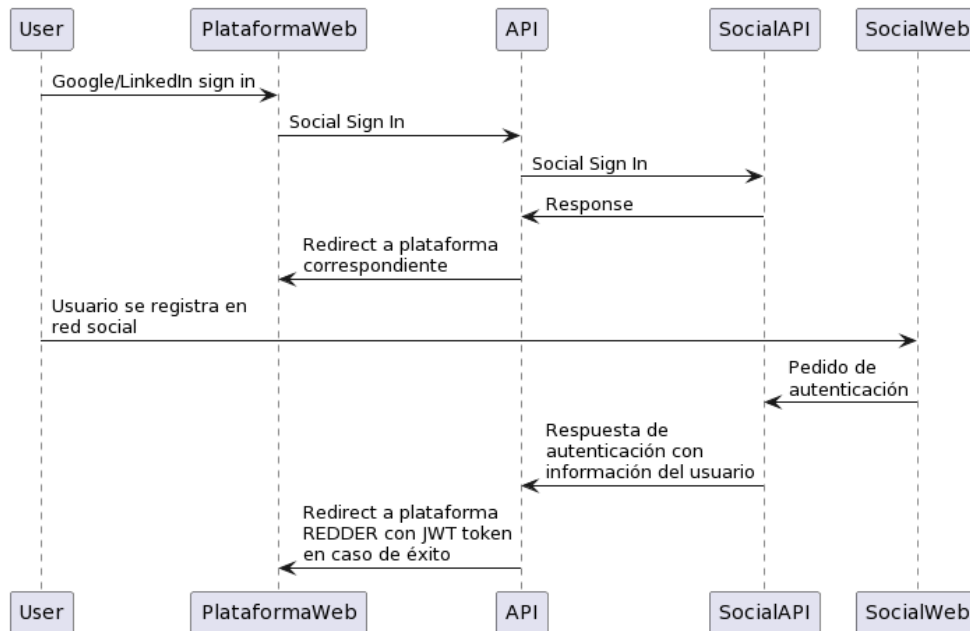


Figura 57: Flujo de autenticación mediante redes sociales

6.3.2. Sendgrid

Como ya se mencionó anteriormente, *Sendgrid* [25] es una plataforma utilizada para el envío de correos electrónicos.

Se decidió utilizar esta herramienta frente a otras también muy conocidas ya que es simple de configurar, el equipo ya tenía experiencia y además la versión gratuita se adapta bien a las necesidades actuales de REDDER. Algunas de las alternativas que contemplamos fueron Sendinblue, Amazon SES, MailChimp y HubSpot.

El equipo de REDDER manifestó el interés de poder enviar correos electrónicos tanto de forma manual como para ciertos eventos. Alguno de los eventos para los cuales se configuró el envío de correos son la confirmación de armado de una pareja y la finalización de la misma.

Por el momento se está utilizando de manera gratuita, pudiendo enviar hasta cien correos diarios, lo cual es suficiente para la plataforma actualmente. Además, se integra al stack de tecnologías de forma relativamente sencilla y

permite crear plantillas de los correos que se quieren enviar de forma muy sencilla sin la necesidad de tener conocimientos de programación.

A continuación se muestra un listado con todas las plantillas generadas y utilizadas por la plataforma para el envío de correos:

TEMPLATE	LAST EDITED
> Partnership Rejection Mail - Redderer	Mar 1, 2022 12:13 PM CST
> Partnership Acceptance Mail - Redderer	Mar 1, 2022 12:08 PM CST
> Rejection Mail - Referrer	Mar 1, 2022 11:55 AM CST
> Partnership Request - Referrer	Feb 28, 2022 1:59 PM CST
> Acceptance Mail - Referrer	Feb 23, 2022 1:33 PM CST
> Welcome Mail - Referrer	Feb 23, 2022 1:07 PM CST
> Welcome Mail - Redderer	Feb 23, 2022 12:47 PM CST
> Recover Password Email	Dec 11, 2021 6:51 AM CST
> Validate Email	Nov 28, 2021 2:58 PM CST

Figura 58: Plantillas en Sendgrid

Estas plantillas son utilizadas por el sistema a través de una API expuesta por Sendgrid para enviar los correos.

Además, Sendgrid presenta un tablero de información indicando la cantidad de correos que se enviaron cada día, indicando cuantos de ellos fueron abiertos y la cantidad de clics en ellos. Esto es algo que aporta mucho valor al equipo de operaciones de REDDER.

6.3.3. Azure

Azure es una de las plataformas más conocidas mundialmente, ya que tiene un sinnúmero de recursos y soluciones disponibles. En este caso, además del despliegue de la aplicación, del cual se hablará en el capítulo 7, se utilizó una integración disponible para el manejo de imágenes.

Como se mostró en la Figura 13, uno de los requisitos que se pidió es que los usuarios tengan una imagen asociada, ya sea obteniéndola de una red social,

o que el usuario pueda subir una que sea de su agrado. En el segundo caso es donde entra esta nueva integración. Con el recurso de almacenamiento en la nube que nos brinda Azure (*Storage Account*) y la posibilidad de subir imágenes al mismo a través de una API, es que se solucionó el requerimiento en cuestión, dándole la posibilidad al usuario de subir una imagen de perfil, actualizarla, o eliminarla. Las imágenes se almacenan en Azure (al igual que el sistema en el entorno de producción, como se explicará en el capítulo 7), y las peticiones para la subida de archivos se hacen siempre desde el *Backend* para preservar las claves privadas que Azure otorga para la subida segura de archivos a sus contenedores. A continuación se presenta un diagrama de secuencia explicando este flujo:

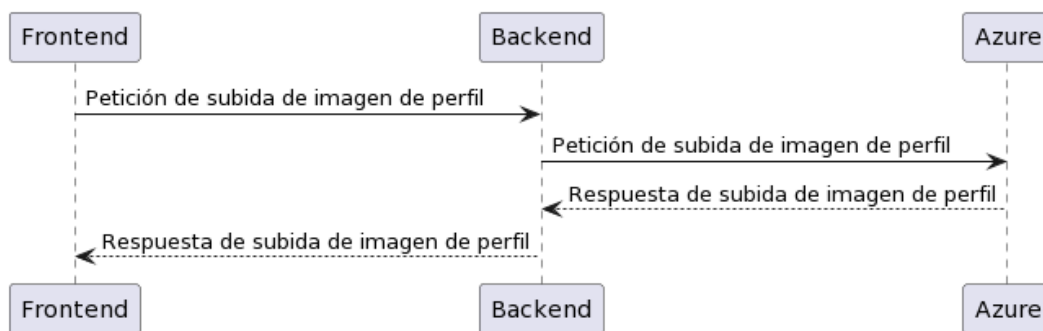


Figura 59: *DSS para subida de imágenes*

6.3.4. Google Analytics

Google Analytics es de las herramientas más útiles y usadas mundialmente [31], donde en su plataforma se pueden ver diferentes datos e información de los usuarios que acceden al sistema, como el país desde donde ingresa a la plataforma, las páginas visitadas y más información demográfica de los usuarios.

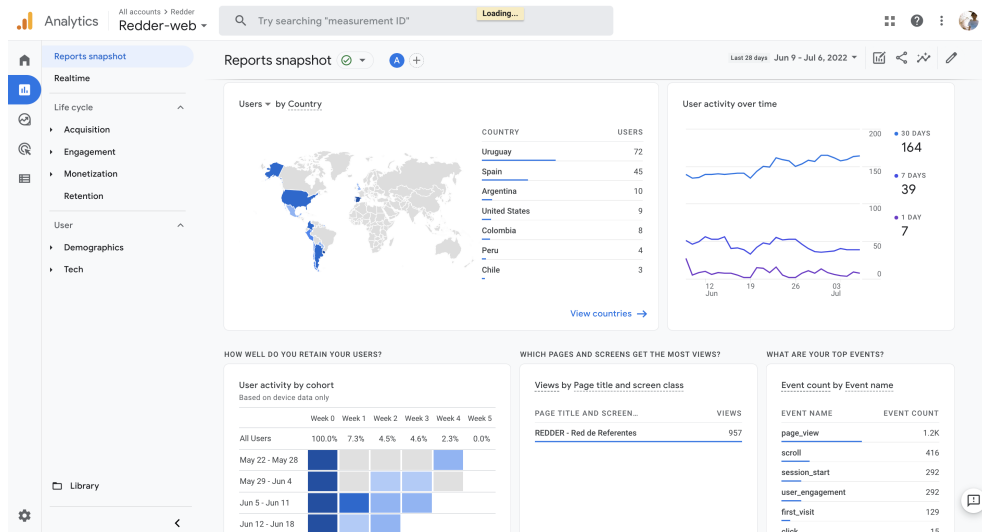


Figura 60: *Analíticas de Google*

Usualmente se usa esta información para tomar decisiones a nivel de producto, como ver cuales son los flujos más utilizados por los usuarios, de dónde son estos usuarios, cuantos de ellos son nuevos y cuantos son recurrentes, entre otros, pudiendo ayudar así a la toma de decisiones estratégicas de la organización.

La información que se obtiene de Google Analytics será utilizada por el equipo de REDDER para entender desde donde y como los usuarios están usando la plataforma. Esto también les será útil a la hora de tomar decisiones sobre las próximas funcionalidades a implementar o mejorar.

7. Despliegue de la solución

Se decidió desplegar el sistema en ambientes de prueba en una etapa temprana con el fin de que fuera fácilmente probable y accesible tanto por el equipo de desarrollo pudiendo hacer las pruebas funcionales en un entorno más similar a lo que sería producción, así como por las integrantes del directorio de REDDER, pudiendo tener feedback de forma fácil y directa.

En este capítulo se describe como fue el proceso de despliegue.

7.1. Netlify

En un principio se decidió utilizar Netlify para el despliegue de las dos aplicaciones de frontend, es decir, la aplicación cliente (REDDER APP) y la aplicación de administración (backoffice).

Netlify es una plataforma que, de forma sencilla permite desplegar y automatizar aplicaciones web estáticas. Conectando la misma con Github, se logró tener despliegues automáticos sin mucha configuración. Es importante destacar que se decidió utilizarla porque el despliegue y la integración continua se hace de forma directa y sencilla, además de tener la posibilidad de que sea gratis, algo importante en esta primer etapa de desarrollo.

7.2. Heroku

Por otro lado, se utilizó Heroku para el despliegue de la aplicación de Backend. Heroku, a diferencia de Netlify, ejecuta las aplicaciones en *'dynos'*, que, en definitiva, son contenedores Linux livianos y aislados. También presenta una posibilidad de hosteo gratis, por lo que entre Netlify y Heroku se tuvo todo el sistema hosteado de forma gratuita, pudiendo tener una versión del sistema corriendo en la nube.

7.3. Azure

Por último, y en una etapa más avanzada del desarrollo, ya casi terminando la etapa de implementación, se consiguió mediante una alianza entre REDDER y Microsoft acceso y uso gratis de Azure.

Por lo que se decidió migrar todo a Azure, ya que brindan un ambiente de producción de forma gratuita en una de la nubes más utilizadas hoy en día.

Como ninguno de los miembros del equipo tenía experiencia en el despliegue de soluciones en la nube de Microsoft, hubo un proceso de estudio para ver cual era la mejor forma de desplegar la solución. Se decidió mover todo el entorno de desarrollo a Microsoft también, para que quede todo centralizado en una misma plataforma. Como consecuencia de este estudio inicial, se encontró el concepto de *'resource group'*, utilizada para agrupar servicios dentro de la plataforma, es por esto, que se crearon dos *'resource groups'* distintos, uno para un entorno de desarrollo, y otro para el entorno de producción.

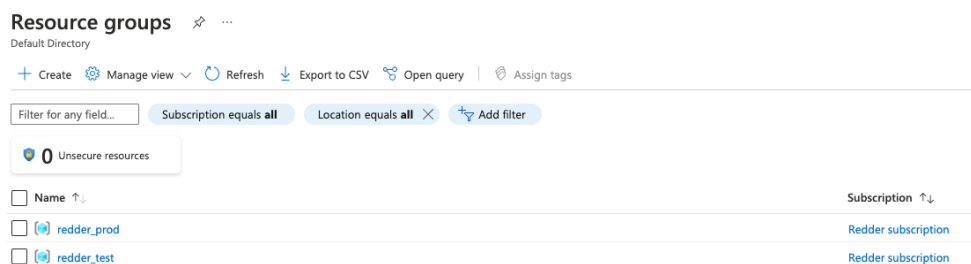


Figura 61: *Resource groups*

Para el despliegue de ambas aplicaciones de Frontend como aplicaciones web estáticas, se utilizó el servicio de *'Azure Storage account'*, ya que en la documentación de Microsoft [32] se recomendaba este recurso en el caso descrito. Además se utilizó otro recurso de estos para las diferentes guías de ayuda que se pueden ver en la plataforma. Es decir, se utilizaron tres *'Azure Storage account'* diferentes, uno para el despliegue de la aplicación web estática, otro para el despliegue de la aplicación de administración, también como una aplicación web estática y una tercera para el guardado de las guías de ayuda mostradas a lo largo del sistema.

Por otro lado, fue necesario configurar el despliegue del Backend. Es claro que el mismo no se puede hacer de la misma manera. Luego de realizar un estudio de la documentación de Microsoft [33], se concluyó que la aplicación de Backend Node se debía desplegar en un App Service. Con esto se tiene todo lo necesario para el despliegue de todo el sistema.

A continuación se muestra esta configuración en Azure, vale la pena mencionar que el entorno de producción es exactamente igual que el utilizado para desarrollo pero con un plan pago, para que la aplicación tenga un mejor desempeño en producción.






Name ↑↓	Type ↑↓
 redder-api-dev	App Service plan
 redder-api-dev	App Service
 redderbotest	Storage account
 reddertest	Storage account
 redderwebsitetest	Storage account

Figura 62: *Servicios de azure*

Por último, pero no menos importante, en producción fue necesario realizar la configuración para tener dominios propios. Es por esto que se investigó la posibilidad de utilizar Azure CDN [34], creando dos 'endpoints' para cada una de las aplicaciones.

Endpoints				
Hostname	Status	Protocol	Origin type	Custom domains
adminredder.azureedge.net	 Running	HTTP, HTTPS	Storage static website	admin.redderglobal.org
redderwebsite.azureedge.net	 Running	HTTP, HTTPS	Storage static website	app.redderglobal.org

Figura 63: *Azure CDN*

De esta manera, se tienen ambos ambientes y todo lo necesario, excepto el despliegue de la base de datos dentro de Azure. Algo que quedó pendiente fue la configuración para el despliegue continuo de las ramas en Github.

7.4. MongoDB Atlas

Las bases de datos, tanto la utilizada en el ambiente de desarrollo como la de producción, están desplegadas en MongoDB Atlas [35]. Se hace de esta manera porque es lo más rápido y sencillo al momento de hacer la configuración, además de que existe también la posibilidad de tener una base de datos desplegada de forma gratuita, utilizada para el ambiente de desarrollo, y una versión paga con más recursos para producción. Además

presenta una interfaz amigable y sencilla para ver los documentos insertados, simplificando así el desarrollo y configuración.

Es claro que teniendo la posibilidad de hacer el despliegue en la nube de Microsoft (Azure) de forma gratuita, se podría pensar en una migración de las bases de datos como trabajo futuro, y, de esta manera, tener absolutamente todo centralizado en Azure, con la ventaja de tenerlo de forma gratuita. De igual manera, habría que estudiar bien qué servicios se necesitan para esto, y si se tienen las mismas funcionalidades que en MongoDB Atlas.

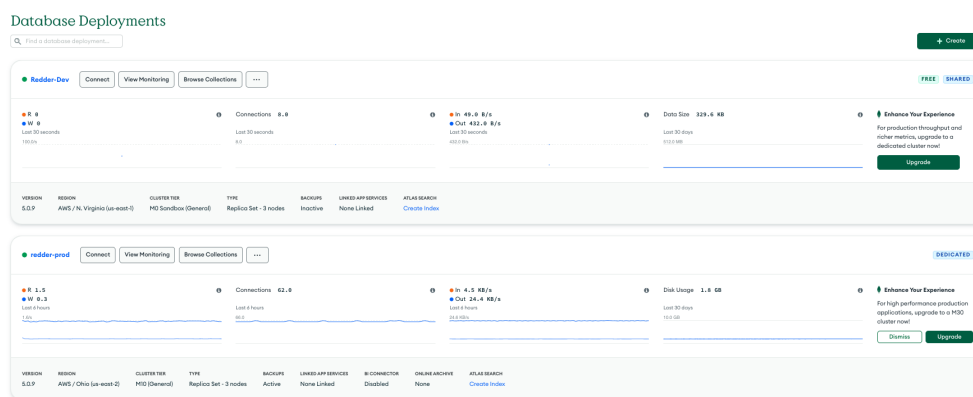


Figura 64: Despliegue en MongoDB Atlas de las bases de datos

8. Pruebas y validaciones

En este capítulo se describen las diferentes pruebas y validaciones que se realizaron tanto a nivel de código, para que el sistema sea mantenible, como a nivel funcional para minimizar las probabilidades de que los usuarios experimenten fallas al usar la plataforma.

8.1. Revisión de código

El proceso de revisión de código se realizó en cada una de las solicitudes de cambios también conocidas como “pull requests” utilizando la herramienta GitHub que se describe en la sección [9.2.1](#).

Como se describe en el proceso de Gitflow en la sección [9.1.2](#) cuando un integrante del equipo de desarrollo va a comenzar a desarrollar una nueva funcionalidad debe crear una rama nueva desde “master”, trabajar en esa rama y cuando considere que el trabajo está finalizado deberá enviar un “pull request” hacia la rama “master”, es en esta solicitud de cambios en donde la otra parte del equipo de desarrollo realiza la revisión de código.

En las figuras [65](#) y [66](#) se pueden observar capturas de revisiones de código en los distintos repositorios de REDDER:

Merged **Add Notes (Observaciones) #53**
 martinsd94 merged 2 commits into master from bugfixes-martin on Mar 21

lucassuburu on Mar 21
 Why we don't delete this?

Reply...

Unresolve conversation martinsd94 marked this conversation as resolved.

```

src/locales/en.json Outdated Hide resolved
... @@ -118,7 +121,7 @@
118 121 "resume_placeholder_redderer": "Explain in a minimum of 250 characters and a maximum of 1000
119 122 "submit": "Submit",
120 123 "confirm": "Confirm",
121 - "no_partnerships_message": "You have no partnership requests pending. While you get some, yo
124 + "no_partnerships_message": "Waiting for Redderers requests, we invite you to see some inform
  
```

lucassuburu on Mar 21
 Maybe While waiting for Redderers requests, ... ?

Figura 65: Revisión de código en REDDER App

Merged **Cancel partnership #15**
 lucassuburu merged 3 commits into master from cancel_partnership on Nov 25, 2021

```

src/apiV1/partnership/partnership.controller.ts Hide resolved
304 + $set: {
305 +   status: PartnershipStatus.canceled,
306 +   description: req.body.reason,
307 +   partnership_end: new Date(),
  
```

martinsd94 on Nov 25, 2021
 maybe cancelled_at is more consistent with the other syntax (cancelled_by)

lucassuburu on Nov 25, 2021 Author
 But maybe was ended due to timing, and was not canceled at all

martinsd94 on Nov 25, 2021
 got it 👍

Reply...

Unresolve conversation lucassuburu marked this conversation as resolved.

martinsd94 approved these changes on Nov 25, 2021 View changes

Figura 66: Revisión de código en REDDER API

En las mismas se puede observar como antes de introducir los cambios de código en la rama principal se realiza una revisión e intercambio de ideas o sugerencias entre los integrantes del equipo.

Este proceso tiene como objetivo minimizar el riesgo de introducir errores de código, código repetido o innecesario y a su vez evitar malas decisiones o malas practicas. [36]

8.2. Pruebas funcionales

Como parte del proceso se decide crear un documento de plan de pruebas en el cual se detallan los flujos principales de la plataforma, como el registro y autenticación de usuarios, la formación de parejas o la cancelación de las mismas. Este documento se puede encontrar adjunto en el [Anexo G](#).

El objetivo de este documento es servir de guía para realizar pruebas de regresión a la hora de publicar una nueva versión de las aplicaciones con nuevas funcionalidades. Verificando que cada uno de los flujos listado en este documento funciona correctamente se esta minimizando el riesgo de introducir nuevos errores al publicar una nueva versión.

También se manejó la posibilidad de automatizar estas pruebas e incluirlas en el flujo de integración continuo utilizando herramientas conocidas como Cypress [37] o Selenium [38] pero se decidió por parte de la organización no implementar estas pruebas y priorizar funcionalidades que se consideran dentro del alcance del prototipo.

8.3. Validación con usuarios

Con el fin de validar las funcionalidades, usabilidad, posibles fallas y mejoras de la plataforma se decidió crear y desplegar un ambiente de desarrollo en donde se publicaban a diario los avances. Los motivos por los cuales se tomó esta decisión fue por un lado para recibir feedback sobre las funcionalidades y detectar oportunidades de mejoras en etapas tempranas del proceso de desarrollo optimizando así los tiempos de todos los involucrados.

En este proceso se invitó a las personas que conforman el directorio de la red y a los integrantes del equipo de desarrollo a crearse usuarios en este ambiente de pruebas para simular un entorno real en donde usuarios con distintos roles (“Redderers” o “Referentes”) interactúen mediante el uso de

la plataforma. Durante estos procesos de pruebas se obtuvieron varias sugerencias y comentarios los cuales fueron muy valiosos para mejorar las funcionalidades y usabilidad del sistema. Algunas de estas sugerencias fueron de diseño (cosas que no se encontraban con facilidad, o que faltaban), pero otras, fueron nuevos requerimientos como la posibilidad de mostrar guías para los usuarios, mejorando de esta forma la plataforma.

También se realizó una encuesta a las siete integrantes del directorio de la red, recavando información sobre el resultado del proyecto. En lo que respecta a la usabilidad de la plataforma el resultado de la encuesta es muy positivo, el 71.4 % respondió con el puntaje máximo de 5 y un 28.6 % con un puntaje de 4.

La plataforma resulta sencilla de usar, incluso para personas no familiarizadas con la tecnología

7 responses

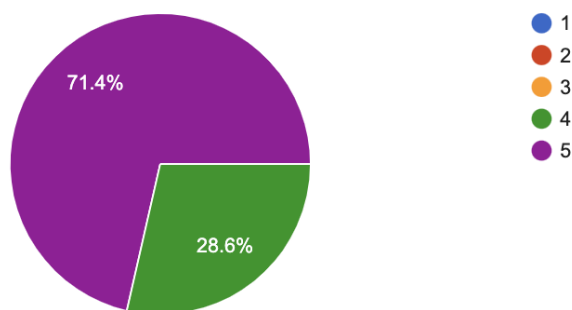


Figura 67: Respuesta pregunta 3

En la pregunta correspondiente al resultado final del proyecto, tuvimos también una muy buena respuesta ya que el 85.7 % con un puntaje de 5 y un 14.3 % con un puntaje de 4.

Estoy conforme con el resultado final de la plataforma

7 responses

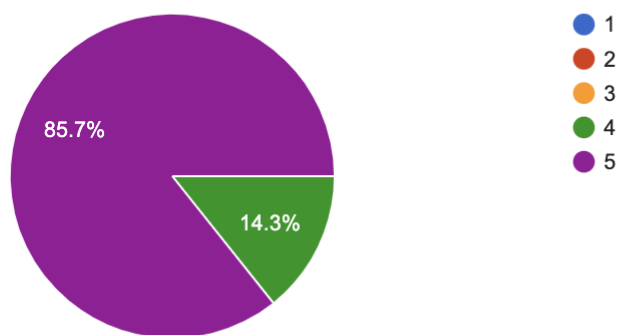


Figura 68: *Respuesta pregunta 4*

El resto de las preguntas y sus resultados se pueden ver en el [anexo C](#)

9. Gestión del proyecto

En este capítulo se describe como se organizó el equipo para llevar a cabo la gestión del proyecto mencionando la metodología de trabajo, las herramientas utilizadas y como fue la comunicación tanto con el equipo de REDDER como con la tutora.

9.1. Metodología

El desarrollo de la plataforma fue realizado con un enfoque iterativo e incremental utilizando metodologías ágiles. Comenzamos utilizando “Scrum” y luego pasamos a utilizar “Kanban” como marcos de trabajo. Este cambio se dio principalmente por la poca disponibilidad de tiempos tanto del equipo de desarrollo como de el directorio de REDDER.

Desde el comienzo del proyecto se tuvo en claro que para el desarrollo de una plataforma de esta escala que será utilizada por usuarios reales y en donde los requerimientos no estaban definidos desde el comienzo era imprescindible una buena gestión del proyecto.

9.1.1. Metodologías Ágiles

Las Metodologías Ágiles son un enfoque iterativo e incremental para la gestión de proyectos y el desarrollo de software que ayuda a los equipos a entregar valor a sus clientes más rápido. En lugar de tener que esperar por un lanzamiento completo en un equipo ágil se entrega el trabajo en incrementos pequeños pudiendo así recibir feedback en etapas tempranas.[39]

Los requisitos, planes y resultados se evalúan continuamente para que los equipos tengan un mecanismo natural para responder rápidamente a los cambios.

Algunas de las características principales del desarrollo de software “Agile” que aportaron valor al proceso fueron tener equipos auto-gestionados, la posibilidad de mostrar avances y recibir feedback en etapas tempranas.

Manifiesto Ágil El Manifiesto Ágil [40] fue escrito en Febrero del 2001 por diecisiete profesionales de software. Si bien los participantes discrepaban en

algunas cosas encontraron consenso en torno a cuatro valores fundamentales.

“Individuos e interacciones sobre procesos y herramientas

Software funcionando sobre documentación extensiva

Colaboración con el cliente sobre negociación contractual

Respuesta ante el cambio sobre seguir un plan

Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda.” [41]

Estos cuatro principios se siguieron durante todo el proceso de desarrollo, donde siempre se priorizó la elaboración del sistema estando abiertos a los cambios que surgieron hasta los últimos momentos del proyecto.

Scrum Es un marco de trabajo simple que promueve la colaboración en los equipos para lograr desarrollar productos complejos de manera incremental. Se basa en la auto-gestión de los equipos para resolver problemas complejos inspeccionando y adaptando continuamente. [42]

Scrum cuenta con algunos eventos para evitar la existencia de reuniones no definidas, las cuales en exceso impactan negativamente en la productividad de los equipos.

Los eventos de Scrum son:

- **Sprint:** Es un contenedor del resto de los eventos de Scrum, generalmente dura entre 2 y 4 semanas
- **Sprint Planning:** Reunión de planificación al comienzo del Sprint
- **Daily Scrum:** Reunión diaria de 15 minutos, generalmente participa solamente el equipo de desarrollo
- **Sprint Review:** Reunión que ocurre al final de Sprint en donde se presenta las nuevas tareas o funcionalidades
- **Sprint Retrospective:** Reunión que ocurre al final de Sprint en donde el objetivo es identificar mejoras para los siguientes Sprints

En el contexto de este proyecto y por disponibilidad de tiempos se realizaban menos reuniones que las estipuladas; a modo de ejemplo, las reuniones “Daily

Scrum” se realizaban una o dos veces por semana en lugar de diarias como se recomienda en el marco de trabajo.

Los artefactos de Scrum que se utilizaron durante el proceso de desarrollo son:

- **Product Backlog:** Es la lista priorizada de funcionalidades que debe contener un producto
- **Sprint Backlog:** Son el conjunto de tareas seleccionadas para el sprint actual
- **Increment:** Es el producto o conjunto de tareas completadas en un sprint

El equipo definió e hizo uso de estos elementos a lo largo del proyecto, para mantener y dar seguimiento a los mismos se utilizaron las herramientas Trello [43] y Notion [44] las cuales se describen en la sección 9.2.

Algo que también se definió en equipo y es muy común al trabajar en un marco de trabajo Scrum fue una “Definition of Done”, esto es que tiene que tener una tarea o historia para poder ser considerada como terminada. En la definición se tuvieron en cuenta que la tarea haya tenido revisión de código, haya sido verificada por el desarrollador mediante test funcional y esté pronta para ser puesta en ambiente de desarrollo o producción.

Como se mencionó anteriormente luego de estar trabajando con Scrum por algunos meses el equipo decidió pasar a “Kanban” ya que Scrum no se adaptaba por la poca disponibilidad de tiempos, tanto por la duración de los Sprint como por la cantidad de reuniones recomendada.

Kanban Kanban es un marco de trabajo muy popular para un desarrollo de software ágil. Requiere una comunicación en tiempo real sobre la capacidad y una total transparencia del trabajo. Los elementos de trabajo se representan visualmente en un “tablero de kanban”, lo que permite a todos los miembros del equipo ver el estado de cada tarea en cualquier momento. [45]

Este marco de trabajo se adaptó mucho mejor a los tiempos y carga de trabajo del equipo de desarrollo y equipo de operaciones de REDDER ya que los tiempos para el desarrollo y planificación eran más flexibles que con Scrum.

Al pasar de utilizar Scrum a Kanban se tuvo que adaptar el tablero de Trello ya que había ciertas columnas como por ejemplo dividir las tareas por Sprint que dejaban de ser útiles. Las columnas que utilizamos en el tablero Kanban son las siguientes:

- **To-Do:** Conjunto de tareas prontas para trabajar ordenadas por prioridad
- **Doing:** Conjunto de tareas que están siendo desarrolladas
- **Code Review:** Conjunto de tareas prontas para revisión de código
- **Testing** Conjunto de tareas prontas para realizar Testing funcional
- **Done:** Conjunto de tareas finalizadas

9.1.2. Control de versiones de código

En este proceso de desarrollo se utilizó Git como software de control de versiones mediante el uso de la herramienta GitHub que se describe más adelante.

Además para unificar la forma en la que se crean las ramas en Git se decidió usar el flujo de trabajo conocido como **Gitflow**. Este es un modelo alternativo de creación de ramas en Git en el que se utilizan ramas de función y varias ramas principales, a cada una de estas se le adjudican funciones muy específicas y se define como deben interactuar entre si. [46]

Una de las grandes ventajas de utilizar Gitflow es que todos los desarrolladores dentro del equipo pueden entender fácilmente la función de cada rama dentro del proyecto.

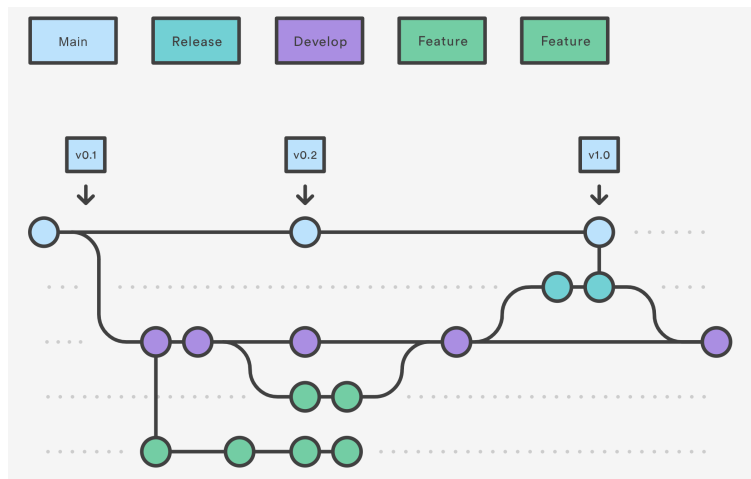


Figura 69: Flujo de trabajo Gitflow

En la Figura 69 se puede observar un diagrama de como funciona este flujo de trabajo. Allí se tiene una rama principal, comúnmente llamada “main” o “master” donde generalmente el código que contiene esta rama es lo que se ejecuta en el ambiente de producción, luego se tiene una rama “develop” donde generalmente el código que contiene esta siendo ejecutado en un ambiente de pruebas o local a cada desarrollador.

A la hora de tener que implementar una nueva funcionalidad cada desarrollador debe crear una nueva rama de “feature” desde la rama “develop”, trabajar en esa rama y una vez que considera que esa funcionalidad esta pronta envía una solicitud de cambios hacia la rama “develop”.

En este flujo de trabajo cuando se quiere publicar una nueva versión se deberá crear un “pull request” desde “develop” hacia “main” y una vez aprobados e introducidos esos cambios generalmente se publica el nuevo código para que este se ejecute en el ambiente de producción.

Este flujo de trabajo fue el que se utilizó por parte del equipo de desarrollo a lo largo de este proyecto y resultó muy útil.

9.2. Herramientas

En esta sección se describen brevemente las herramientas que se utilizaron para facilitar el proceso de desarrollo

9.2.1. Gestión del proyecto

Notion Al comienzo del proyecto se decidió usar Notion ya que es una herramienta para gestión de proyectos que esta siendo muy utilizado hoy en día por startups de software.

Lo utilizamos principalmente para tomar notas de reuniones con la tutora y el equipo de REDDER así como también para crear un tablero en donde se creaban historias de usuario de las tareas que se tenían que implementar.

El tablero que se utilizó constaba de cinco columnas: *Backlog*, *To-Do*, *In Progress*, *Review*, *Completed*. Estas eran muy útiles para identificar de forma rápida e intuitiva el estado actual de cada tarea. Además, en cada historia de usuario se podía agregar la *prioridad* de esa tarea y que aplicaciones dentro del sistema estaban relacionadas con esa tarea.

En las figuras 70 y 71 se pueden observar capturas del tablero y historias de usuario:

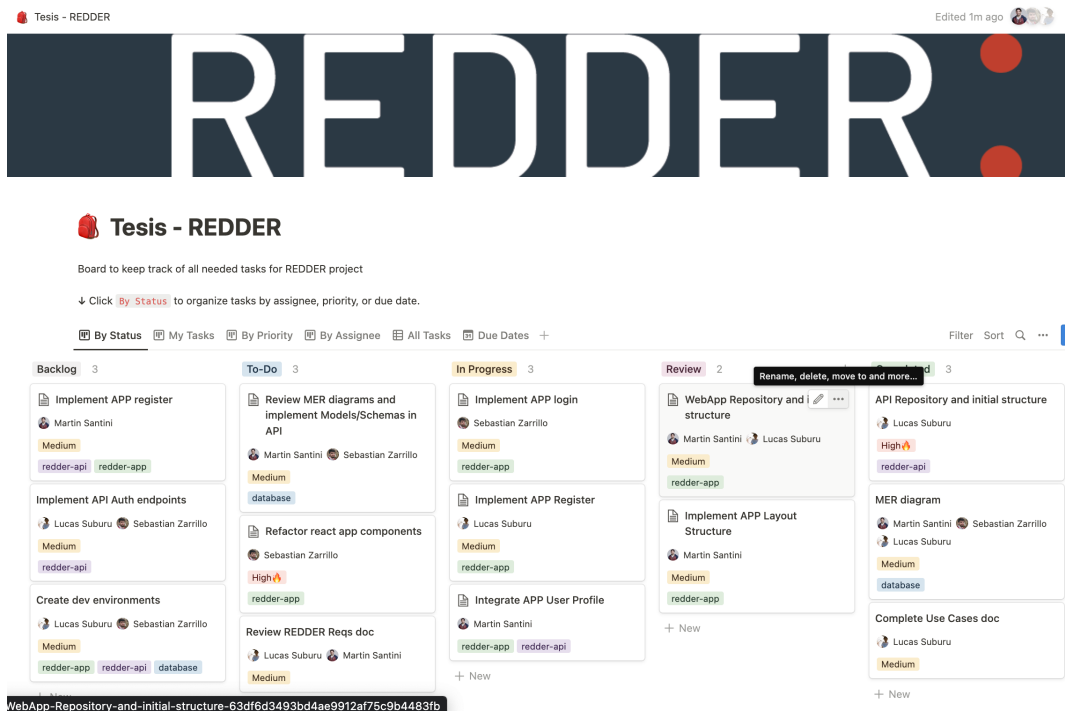


Figura 70: Tablero Kanban en Notion

Refactor react app components

The screenshot shows a Notion task card with the following details:

- Assign:** Martin Santini
- Status:** To-Do
- Priority:** High (with a fire icon)
- Date Created:** August 16, 2021 10:54 PM
- Repository:** redder-app
- Actions:** + Add a property, + Add a comment...

Description

Refactor `redder-app` components removing all `hardcoded` data and making the components more reusable

+ ⓘ **Note:** this task will probably need to be divided into smaller tasks

Acceptance Criteria

- There shouldn't be hardcoded data
- Components should be reusable

Figura 71: *Historia de usuario en Notion*

Luego de algunas semanas trabajando usando Notion el equipo de desarrollo decidió pasar a usar Trello ya que resultaba más intuitivo y sencillo de utilizar.

Trello Trello es una herramienta visual que permite a los equipos gestionar cualquier tipo de proyecto y flujo de trabajo, así como supervisar tareas. [43]

Al igual que al comienzo con Notion, esta herramienta se utilizó para hacer seguimiento de las tareas y llevar registro de las reuniones que se llevaban a cabo con REDDER y la tutora.

A continuación se puede observar el tablero que utilizó el equipo en Trello y una historia de usuario a modo de ejemplo:

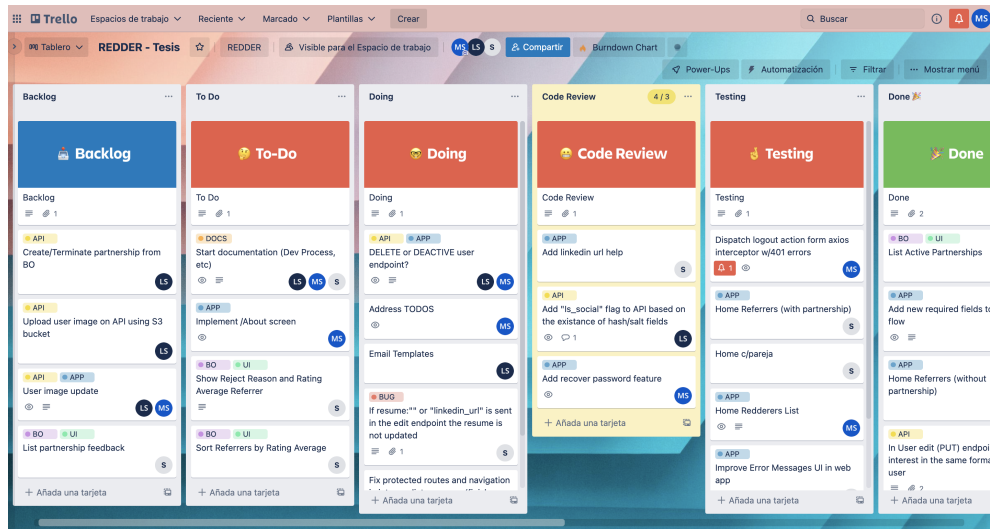


Figura 72: Tablero Kanban en Trello

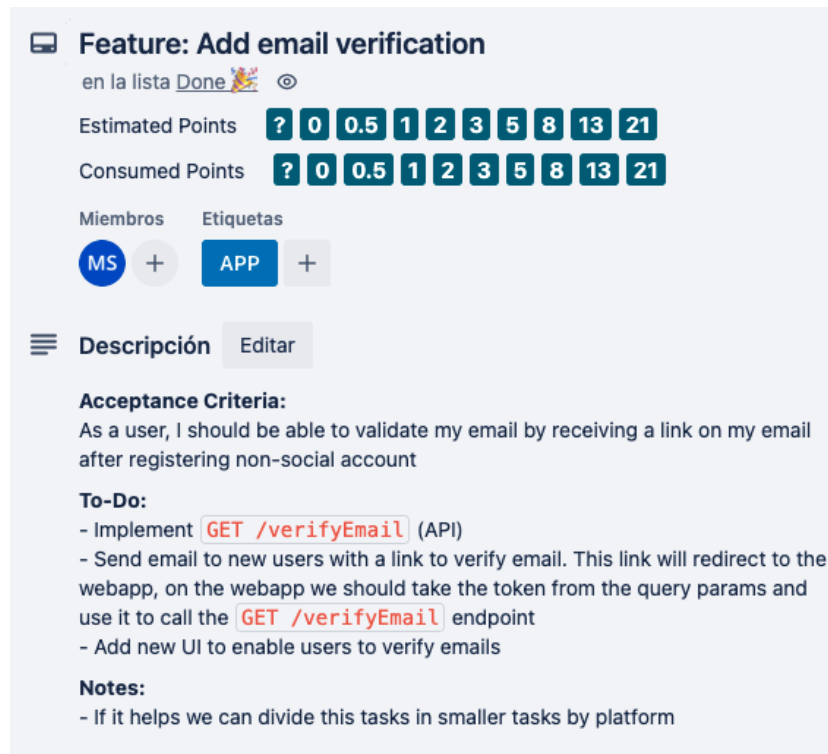


Figura 73: Historia de usuario en Trello

Como se puede observar en las figuras 72 y 73, se utilizaron varias etiquetas para diferenciar las tareas por plataforma (“APP”, “API” o “BO”) así como también por tipo de tarea (“Feature”, “Bug”, “Docs”, “Doubt”). Esta clasificación fue muy útil ya que ayudaba identificar a simple vista el tipo de tarea.

Esta herramienta fue utilizada durante casi todo el desarrollo del sistema y fue muy útil para hacer seguimiento de cada una de las tareas

Google Drive Google Drive es un servicio de alojamiento de archivos en la nube utilizado y reconocido mundialmente.

En el proceso esta herramienta se utilizó para diversos fines, entre ellos la creación y almacenamiento de archivos, documentos y diagramas que se debían compartir con el equipo de REDDER y la tutora de forma sencilla.

En esta herramienta se implementaron los documentos de casos de uso para la plataforma y el backend así como también un documento con un “Roadmap” en el cual se describían las funcionalidades a alto nivel y fechas estimadas de entrega, este documento fue muy útil para manejar las expectativas del equipo de REDDER.

En la Figura 74 se puede observar el *Google Drive* que el equipo utilizó durante este proceso.

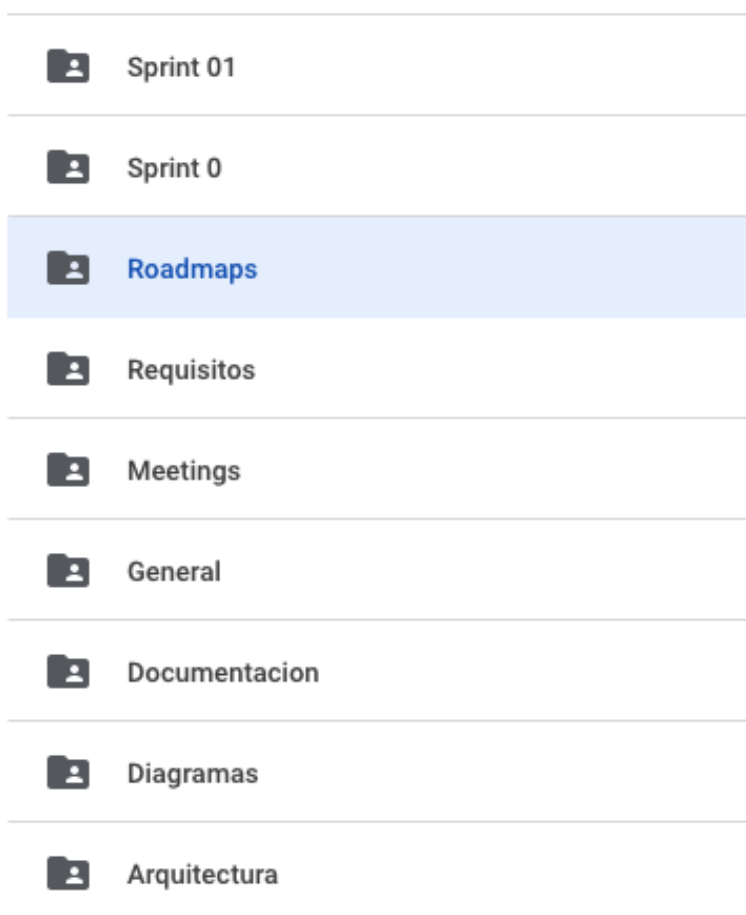


Figura 74: *Google Drive del equipo de Desarrollo*

GitHub GitHub es una plataforma de alojamiento, propiedad de Microsoft, que ofrece a los desarrolladores la posibilidad de crear repositorios de código

y guardarlos en la nube de forma segura, usando un sistema de control de versiones llamado Git. [47]

Se utilizó esta herramienta para alojar el código de los tres repositorios que fueron necesarios a lo largo del proyecto. Para organizarlo de la manera más simple y clara posible se decidió crear una organización dentro de GitHub con el nombre “RedderUY” ya que esto facilitaría el traspaso de código así como también agregar nuevos desarrolladores a la organización en caso que fuese necesario. A continuación se puede observar una imagen de la organización y los repositorios que fueron creados en la misma:

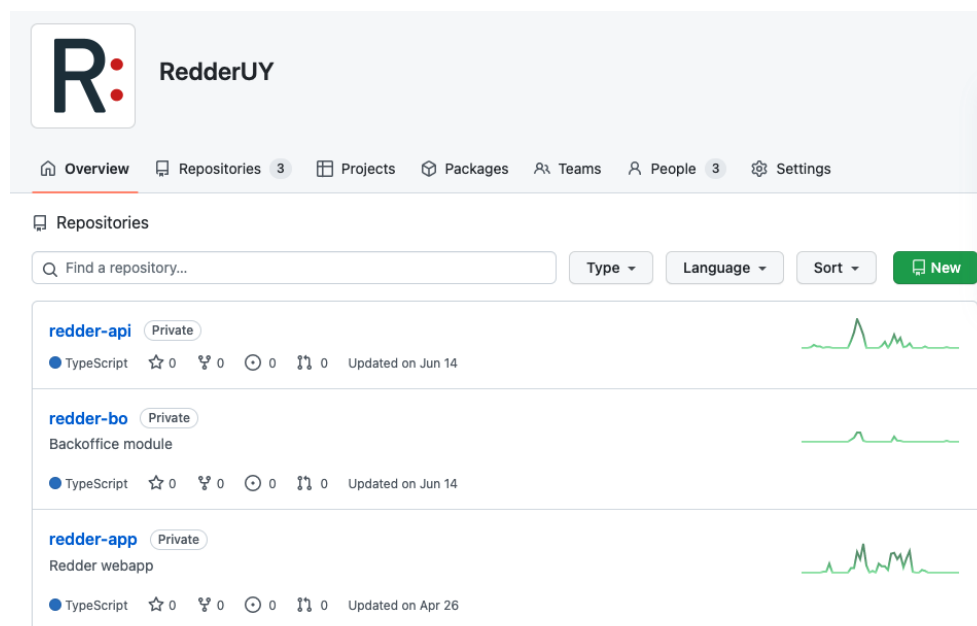


Figura 75: Organización RedderUY en GitHub

En esta herramienta también se realizaron las revisiones de código antes de introducir los cambios en los distintos repositorios, este proceso constaba de crear un “Pull Request” una vez el desarrollador encargado de cierta tarea la consideraba como terminada y asignaba a otro integrante del equipo para que revise el código.

Un “Pull Request” es la creación de una solicitud de cambios y validación de código que se va a introducir en cierta rama de un repositorio. A continuación

se observan una imágenes con “Pull Requests” de ejemplo en la App y API de REDDER:

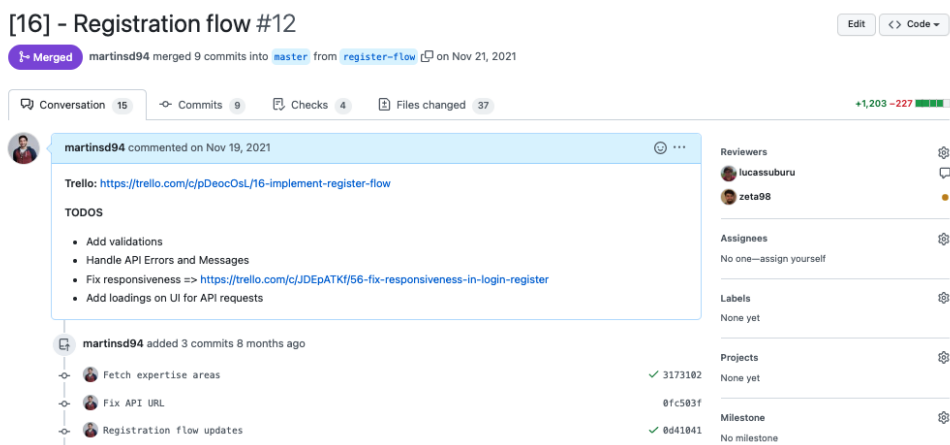


Figura 76: Pull Request en REDDER App

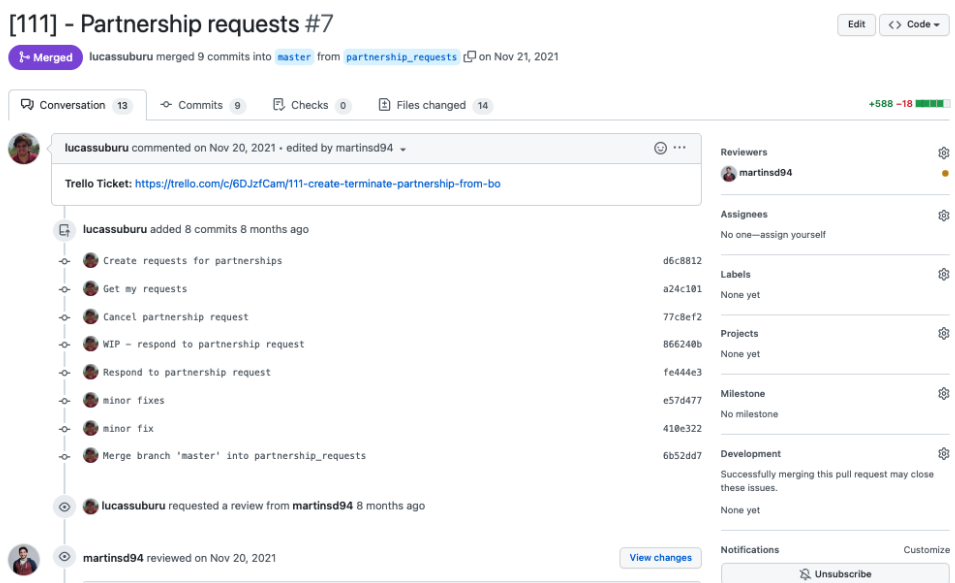


Figura 77: Pull Request en REDDER API

Como se puede observar en los Pull Requests de las figuras 76 y 77, en los mismos se agregaba en el identificador de la tarea en el tablero de Trello en

el título y en la descripción un link a esa tarea seguida de algún comentario si fuese necesario. Esto ayudó mucho a que quienes hicieron la revisión de código para encontrar fácilmente los requerimientos de las tareas vinculadas a esos cambios en el código.

9.2.2. Comunicación

Con el fin de mantener una comunicación fluida tanto con el equipo de REDDER como con la tutora y entre los integrantes del equipo de desarrollo se utilizaron varias herramientas.

A continuación se listan y describen brevemente las mismas.

Slack Slack es una herramienta de mensajería muy utilizada por empresas del sector tecnológico. El equipo de desarrollo decidió utilizar esta herramienta ya que la utilizan a diario en un contexto laboral.

En Slack se pueden crear organizaciones y dentro de las mismas varios “channels” con miembros los cuales son muy parecidos al concepto de “grupo” de otras aplicaciones similares. En el contexto de este proyecto se decidió crear una organización con el nombre de “Redder” y se invitó a formar parte de la misma a los tres integrantes del equipo de desarrollo y a una integrante de REDDER que era la encargada de las decisiones de producto en ciertos momentos del proyecto.

A continuación se puede observar una imagen de la organización en Slack y los distintos “channels” utilizados:

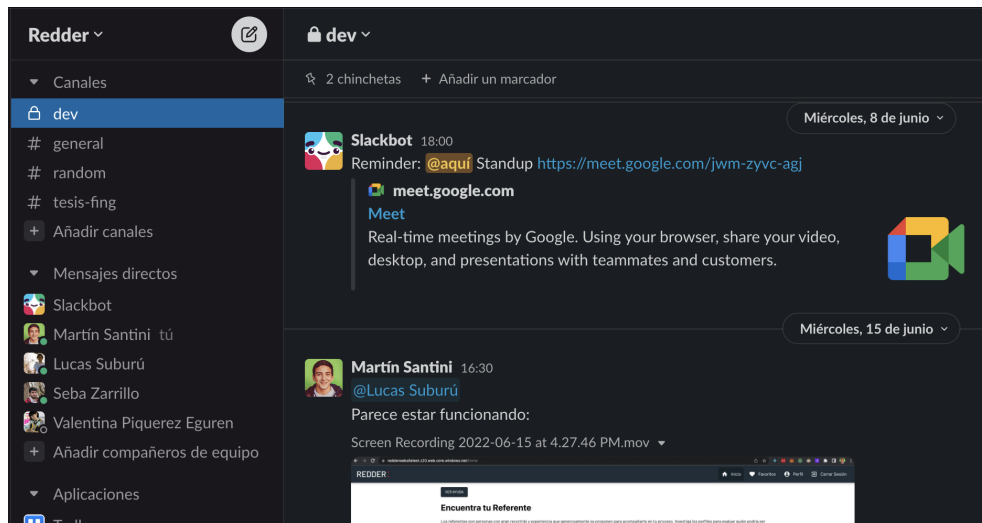


Figura 78: *Slack de Redder*

La única desventaja de utilizar esta herramienta fue que las integrantes del directorio de REDDER no estaban familiarizadas ni utilizaban a diario esta herramienta. Igualmente, fue el canal de comunicación principal entre los integrantes del equipo de desarrollo.

Gmail Es el servicio de correo electrónico de Google.

El mismo fue utilizado a lo largo de todo el proyecto para el intercambio de correos con la tutora y con el equipo de REDDER.

Google Meet Google Meet es uno de los servicios de videoconferencias de Google más utilizado.

En el contexto de este proyecto se utilizó a diario para las reuniones virtuales entre el equipo de desarrollo y con la tutora o el directorio de REDDER.

Zoom Zoom al igual que Google Meet es una herramienta de videoconferencias muy utilizada hoy en día.

Fue utilizado en forma análoga a Google Meet para ciertas reuniones durante el proyecto.

9.3. Resumen

La buena gestión del proyecto no solamente ayudó a que los objetivos planteados desde un comienzo se cumplieran sino que también facilitó a que el equipo de REDDER y la tutora pudieran tener en todo momento acceso a los avances obtenidos de la plataforma y estar informados sobre las próximas funcionalidades en que trabajaría el equipo de desarrollo.

También ayudó a que cada integrante del equipo tanto de desarrollo como de la organización tenga claro en todo momento en que tareas trabajar.

A su vez el hecho de utilizar herramientas que tanto el equipo de desarrollo, como el directorio de la red y la tutora utilizan a diario en otros ámbitos ayudó a tener una comunicación fluida y eficiente entre todas las partes.

Para planificar las distintas etapas del proyecto se utilizaron los diagramas de Gantt [48]. Estos sirven para programar tareas en un largo período de tiempo. Una de sus grandes ventajas es poder visualizar de manera sencilla y realizar un seguimiento de las mismas. De esta manera, se podrá contrastar lo planificado contra lo que termino sucediendo realmente.

A continuación se presenta el diagrama de Gantt creado al inicio del proyecto para las tareas planificadas:



Figura 79: Gantt de las tareas planificado

Por otro lado, se presenta el diagrama de Gantt creado al finalizar el proyecto, es decir, con lo que termino sucediendo realmente:



Figura 80: Diagrama de Gantt de cómo se ejecutaron las tareas realmente

Como se puede observar, el proyecto culminó unos meses después de lo planificado, siendo la tarea de *Diseño de la arquitectura* la principal responsable, ya ésta se prolongó un poco más de lo que se pensaba al principio, repercutiendo así en el resto de las tareas. Éstas se pudieron realizar de acuerdo a lo estimado, pero igualmente se atrasó el proceso unos meses en el tiempo.

10. Conclusiones y trabajo futuro

10.1. Conclusiones

El objetivo principal de este proyecto, como se ha mencionado a lo largo del documento, era construir una plataforma que le brinde a la ONG REDDER (Red de Referentes) la posibilidad de digitalizar procesos e interacciones entre los distintos usuarios que de otra forma serían realizados de forma manual. A su vez esta plataforma ayudará a que la organización llegue a más personas y crezca rápidamente lo cual de ser estos procesos manuales sería inviable.

Como parte del proyecto, se construyó un prototipo de la aplicación para validar la idea. En una primer instancia, el sistema permite la conformación de parejas de trabajo entre usuarios que compartan los mismos intereses, por medio de las ya explicadas áreas de interés, además de permitir el seguimiento de éstas parejas, brindando la posibilidad de registrar reuniones, objetivos y avances sobre los mismos. Sumado a todo esto, se implementó un sistema de administración, dándole la posibilidad a las integrantes de la red de manejar y administrar todo el sistema, desde los usuarios hasta las parejas, además de poder visualizar de forma sencilla diferentes aspectos que pueden ser de utilidad para tomar decisiones de negocio.

Además de todos los casos de uso que se describieron extensivamente en el [capítulo 4](#) también se recabaron algunos requisitos no funcionales que son esenciales para que el sistema pueda ser utilizado en un entorno real. Para el cumplimiento de estos, se decidió tomar el tiempo suficiente para el diseño de la plataforma, usar tecnologías modernas y ampliamente usadas por la comunidad, además de aplicar buenas practicas de código, haciendo especial hincapié en la mantenibilidad y la escalabilidad del sistema, ya que se espera que él mismo se siga desarrollando en un futuro cercano.

REDDER tuvo su [primer encuentro presencial](#) el pasado 1 y 2 de abril de 2022 en Maldonado, Uruguay [49]. En este evento se presento la plataforma que luego saldría a producción. Es por esto que se solicito un [video](#) de presentación de la plataforma, que también sirve como manual de usuario [50]. Es importante mencionar que la plataforma y todas las herramientas que esta utiliza, cumplen con la licencia MIT (Massachusetts Institute of Technology) [51], una licencia permisiva que impone muy pocas limitaciones en su reutilización y uso.

Sin embargo, vale la pena mencionar algunos puntos que nos gustaría haber tenido el tiempo suficiente para poder incluirlos en este proyecto, como pueden ser las pruebas automatizadas en todas las funcionalidades desarrolladas debido a que se decidió priorizar las funcionalidades requeridas y la revisión de código por sobre las mismas. En la siguiente sección se detallaran otros puntos que creemos interesantes como trabajo a futuro para la plataforma.

Como conclusión general, consideramos que el proyecto cumplió con las expectativas ya que se implementaron las funcionalidades y requisitos más importantes, se desplegó el sistema en un entorno de producción y el mismo está siendo utilizado por usuarios reales. Esto está respaldado por una encuesta realizada a las 7 integrantes del directorio de la red, recavando información sobre el resultado del proyecto, se pueden ver las respuestas de esta encuesta en el [anexo C](#)

Es claro que el impacto del sistema fue muy grande, ya que permite a REDDER llegar a diferentes usuarios de todo el mundo. Al día de hoy (*18 de setiembre de 2022*) el sistema presenta unos 110 usuarios activos con 16 parejas activas. Si bien este número no es muy alto, se cree que en el futuro cercano va a aumentar significativamente la cantidad de usuarios y parejas activas.

10.2. Trabajo futuro

Hay varios puntos en donde el sistema puede mejorar o implementar nuevas funcionalidades que, en algún momento del relevamiento de requisitos o del desarrollo se tuvieron en cuenta, pero se dejaron para posterior implementación por tema tiempo y recursos. A continuación se describen algunos de ellos, ordenados según su relevancia e impacto en la organización.

Una posible **migración de la base de datos**, actualmente desplegada en *MongoDB Atlas*, como se mencionó en el capítulo 7 a *Azure* para aprovechar al máximo el patrocinio de Microsoft y reduciendo así costos de despliegue, puede ser de los puntos más importantes y relevantes en cuanto al impacto que tendría en el corto plazo en la organización.

Por otro lado, los **test automatizados** permiten hacer pruebas de regresión fácilmente, y son un recurso muy importante para seguir las buenas practicas

de desarrollo. Creemos que es algo que puede agregar robustez a la aplicación, por lo que puede ser algo a desarrollar en un futuro.

Siguiendo con las funcionalidades que en algún momento del proyecto se conversaron con el directorio de la red para ser implementados pero que por falta de tiempo no fueron incluidos, se pueden mencionar:

- ***Chat de mensajería instantánea***, pudiendo dar la posibilidad de que las Redders y Referentes se comuniquen por medio de la misma plataforma y no por fuera de la misma.
- ***Video conferencias***, siguiendo en la misma línea del punto anterior, dar la posibilidad de tener reuniones virtuales por medio de la plataforma, integrándose con alguna plataforma ya existente pero facilitando su uso a través del mismo sistema y no dejando a los usuarios gestionar los mismos.
- ***Notificar usuarios de Backoffice***, al momento de agregar o dar de baja usuarios de la plataforma de administración, notificar a los mismos vía correo electrónico para que puedan saber
- ***Calendario integrado***, ya sea el calendario de *Google*, de *Outlook* o ambos, permitiendo a los usuarios organizar los encuentros virtuales y seguimientos de forma más sencilla y de forma automática.

Por último, pero no menos importante, podemos mencionar la posibilidad de agregar ***Machine Learning para matcheo de parejas***, siendo este módulo un claro candidato para esto, pudiendo dar muchos más beneficios y dejando que las sugerencias de emparejamiento de los perfiles se hagan de forma más inteligente.

Referencias

- [1] ONU. *Gender Equality*. Última visita Abril 2022. URL: <https://www.un.org/es/global-issues/gender-equality>.
- [2] Cepal. *Progreso y evolución de la inserción de la mujer en actividades productivas y empresariales en América del Sur*. Última visita Setiembre 2022. URL: https://www.cepal.org/sites/default/files/publication/files/42031/RVE122_Avolio.pdf.
- [3] Redder. *Redder Global*. Última visita Julio 2022. URL: <https://www.redderglobal.org>.
- [4] GrowthMentor. *About GrowthMentor*. Última visita Marzo 2022. URL: <https://www.growthmentor.com/about/>.
- [5] MicroMentor. *MicroMentor - Preguntas Frecuentes*. Última visita Mayo 2022. URL: <https://www.micromentor.org/faq/>.
- [6] MentorCloud. *How MentorCloud Works*. Última visita Mayo 2022. URL: <https://www.micromentor.org/faq/>.
- [7] RedME. *RedME - Inicio*. Última visita Mayo 2022. URL: <https://redme.org.uy/>.
- [8] HaceLaFuerza. *HLF - HaceLaFuerza*. Última visita Mayo 2022. URL: <https://www.hacelafuerza.org/>.
- [9] Balsamiq. *Balsamiq*. Última visita Febrero 2022. URL: <https://balsamiq.com/>.
- [10] Ian Sommerville. *Ingeniería de Software*. Pearson Education Limited, 2018.
- [11] ISO 25000. *Mantenibilidad*. Última visita Marzo 2022. URL: <https://iso25000.com/index.php/normas-iso-25000/iso-25010/26-mantenibilidad>.
- [12] Fing. *Escalabilidad*. Última visita Mayo 2022. URL: https://eva.fing.edu.uy/pluginfile.php/243156/mod_resource/content/2/Dise%C3%B1o%20-%20Clase%203.pdf.
- [13] El País. *El 91,5 % de los internautas ya accede a Internet a través del móvil*. Última visita Agosto 2022. URL: <https://elpais.com/tecnologia/2020-03-05/el-915-de-los-internautas-ya-accede-a-internet-a-traves-del-movil.html>.
- [14] *Client and server architecture*. Última visita Abril 2022. URL: <https://es.wikipedia.org/wiki/Cliente-servidor>.

- [15] *Gartner Group*. Última visita Marzo 2022. URL: <https://www.gartner.com/en/information-technology/glossary/enterprise-architecture-ea>.
- [16] *Agenda*. A *light-weight job scheduling library for Node.js*. Última visita Enero 2022. URL: <https://www.npmjs.com/package/agenda>.
- [17] *MERN Stack*. Última visita Mayo 2022. URL: <https://www.mongodb.com/mern-stack>.
- [18] *Typescript*. A *strongly typed programming language*. Última visita Enero 2022. URL: <https://www.typescriptlang.org/>.
- [19] *ReactJS*. Última visita Julio 2022. URL: <https://reactjs.org/>.
- [20] *Redux*. A *Predictable State Container for JS Apps*. Última visita Enero 2022. URL: <https://redux.js.org/>.
- [21] *Axios*. *HTTP client for the browser and node.js*. Última visita Febrero 2022. URL: <https://axios-http.com/>.
- [22] *Express*. *Fast, unopinionated, minimalist web framework for Node.js*. Última visita Enero 2022. URL: <https://expressjs.com/>.
- [23] *Microsoft*. *¿Qué es REST?* Última visita Setiembre 2022. URL: <https://learn.microsoft.com/es-es/azure/architecture/best-practices/api-design#what-is-rest>.
- [24] *Passport*. *Simple, unobtrusive authentication for Node.js*. Última visita Febrero 2022. URL: <https://www.passportjs.org/>.
- [25] *Sendgrid*. *Twilio Email Service*. Última visita Febrero 2022. URL: <https://sendgrid.com/>.
- [26] *Node Schedule*. Última visita Marzo 2022. URL: <https://www.npmjs.com/package/node-schedule>.
- [27] *Node Cron*. Última visita Marzo 2022. URL: <https://www.npmjs.com/package/node-cron>.
- [28] *Bottleneck*. Última visita Marzo 2022. URL: <https://www.npmjs.com/package/bottleneck>.
- [29] *Bull*. Última visita Marzo 2022. URL: <https://www.npmjs.com/package/bull>.
- [30] *Redis*. Última visita Julio 2022. URL: <https://redis.com/>.
- [31] *w3techs*. *Market share trends for traffic analysis tools*. Última visita Agosto 2022. URL: https://w3techs.com/technologies/history_overview/traffic_analysis.
- [32] *Microsot*. *Despliegue aplicación web estática*. Última visita Marzo 2022. URL: [95](https://docs.microsoft.com/en-</div><div data-bbox=)

- us/azure/storage/blobs/storage-blob-static-website.
- [33] Microsoft. *Despliegue aplicación NodeJs*. Última visita Marzo 2022. URL: <https://docs.microsoft.com/en-us/azure/app-service/quickstart-nodejs?tabs=linux&pivots=development-environment-vscode>.
 - [34] Microsoft. *Azure CDN*. Última visita Marzo 2022. URL: <https://docs.microsoft.com/en-us/azure/storage/blobs/static-website-content-delivery-network>.
 - [35] MongoDB. *MongoDB Atlas Cloud*. Última visita Febrero 2022. URL: <https://www.mongodb.com/cloud>.
 - [36] Code Review. Última visita Marzo 2022. URL: <https://www.atlassian.com/agile/software-development/code-reviews>.
 - [37] Cypress. Última visita Julio 2022. URL: <https://www.cypress.io/>.
 - [38] Selenium. Última visita Julio 2022. URL: <https://www.selenium.dev/>.
 - [39] Atlassian. *The Agile Coach*. Última visita Enero 2022. URL: <https://www.atlassian.com/agile>.
 - [40] Agile Alliance. *The Agile Coach*. Última visita Enero 2022. URL: <https://www.agilealliance.org/agile101/the-agile-manifesto>.
 - [41] Agile Manifesto. *Agile Manifesto*. Última visita Marzo 2022. URL: <https://agilemanifesto.org/i>.
 - [42] Scrum Org. *Que es Scrum?* Última visita Enero 2022. URL: <https://www.scrum.org/resources/blog/que-es-scrum>.
 - [43] Trello. *Trello Tour*. Última visita Junio 2022. URL: <https://trello.com/tour>.
 - [44] Notion. *Notion*. Última visita Febrero 2022. URL: <https://www.notion.so/>.
 - [45] Atlassian. *Kanban Atlassian*. Última visita Enero 2022. URL: <https://www.atlassian.com/es/agile/kanban>.
 - [46] Atlassian. *Flujo de trabajo Gitflow*. Última visita Febrero 2022. URL: <https://www.atlassian.com/es/git/tutorials/comparing-workflows/gitflow-workflow>.

- [47] Platzi. *Qué es GitHub y cómo usarlo para aprovechar sus beneficios*. Última visita Febrero 2022. URL: <https://platzi.com/blog/que-es-github-como-funciona/>.
- [48] *Diagrama de Gantt*. Última visita Junio 2022. URL: <https://obsbusiness.school/int/blog-project-management/diagramas-de-gantt/que-es-un-diagrama-de-gantt-y-para-que-sirve>.
- [49] *REDDER - Primer encuentro presencial*. Última visita Agosto 2022. URL: https://www.linkedin.com/posts/redderglobal_reddereferentes-liderazgo-liderazgofemenino-activity-6929788830770188288-VvuA/.
- [50] *REDDER - Video*. Última visita Agosto 2022. URL: https://drive.google.com/file/d/1rfff3hkmRLGfcDvdcn-xo8HOEfd9_GOP/view?usp=sharing.
- [51] *Licencia MIT*. Última visita Agosto 2022. URL: <https://opensource.org/licenses/MIT>.

A. Documento sobre áreas de interés

En este anexo se agrega el documento que envió REDDER para la definición de las áreas de interés y sus correspondientes matcheos

REDDERERS		MATCHEA CON			
		1a opción	2a opción	3a opción	4a opción
Carrera profesional (corporación o empresa)					7
Profesional independiente	Ciencia / Tecnología		1.1	4	7
	Arquitectura		1.1		7
	Administración / Legales		1.1	1.3	7
	Medicina / Estética	2	1.1		7
Tengo una startup / pyme	Estrategia / Ventas / Internacionalización			1.1	7
	RRHH			1.2	7
	Legales / Protección patentes y marcas			1.3	7
	Inversión			1.4	7
Carrera deportiva			2	7	
Arte	Desarrollo de carrera dentro del arte			3	7
Carrera en la ciencia				4	7
Carrera política				5	7
Soy mujer rural				6	7

Las Redderers pueden filtrar previamente p idioma
 locación geográfica
 Hechos los filtros se ven las parejas posibles de acuerdo al orden de opciones de arriba

Figura 81: Mapeo Áreas de Interés

REFERENTES			
1	Empresa	Estrategia / Ventas / Internacionalización	1.1
		RRHH	1.2
		Legales / Protección patentes y marcas	1.3
		Inversión	1.4
2	Deporte / Ciencias de la salud		
3	Arte		
4	Ciencia & Tecnología		
5	Política		
6	Ámbito Rural		
7	Desarrollo personal		
OBSERVACIONES			
<p>Todas las referentes debería poder clicar "7 - Desarrollo personal" Y adicionalmente los campos de expertise</p>			

Figura 82: *Mapeo Áreas de Interés*

B. Maquetas de Balsamiq

En este anexo se puede ver los diseños que se hicieron en Balsamiq. Es importante mencionar que se usaron como puntapié inicial para empezar a discutir requerimientos con los integrantes de REDDER

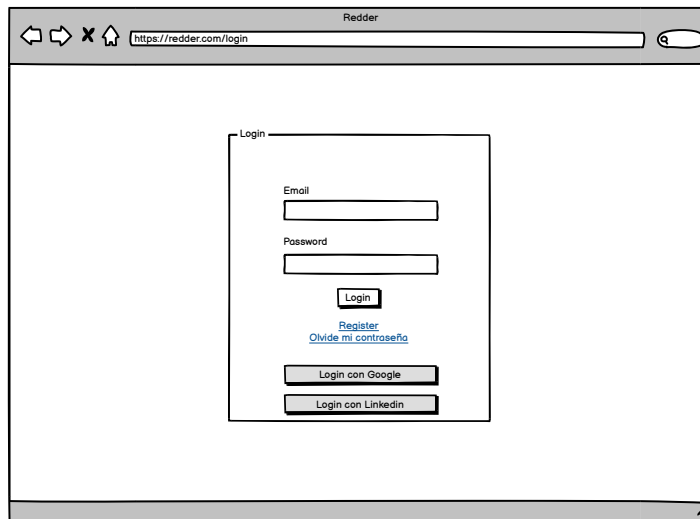


Figura 83: *Maqueta Login*

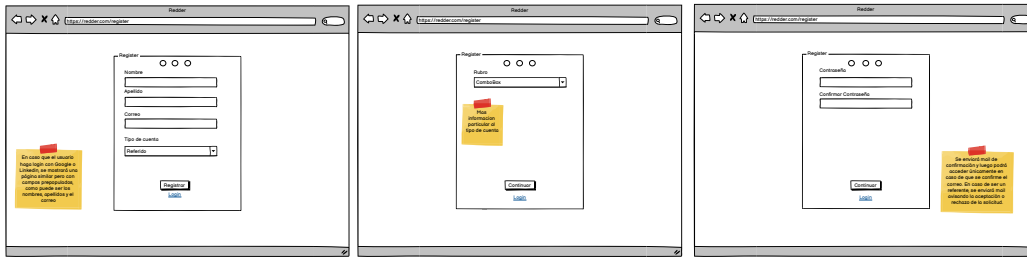


Figura 84: *Maqueta Registro*

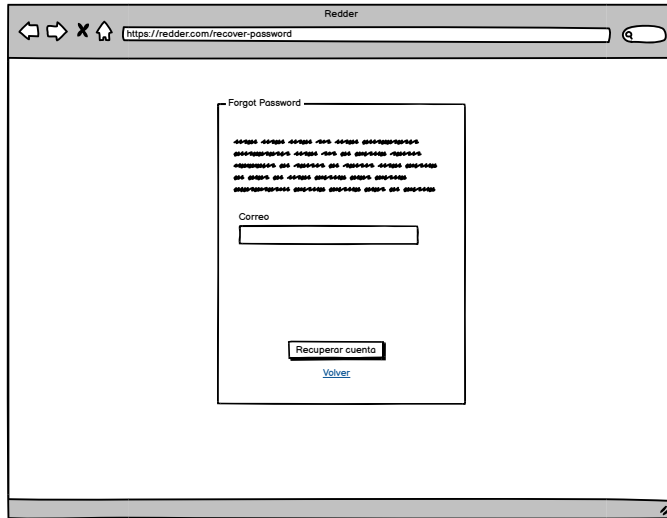
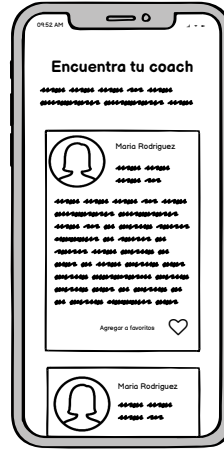
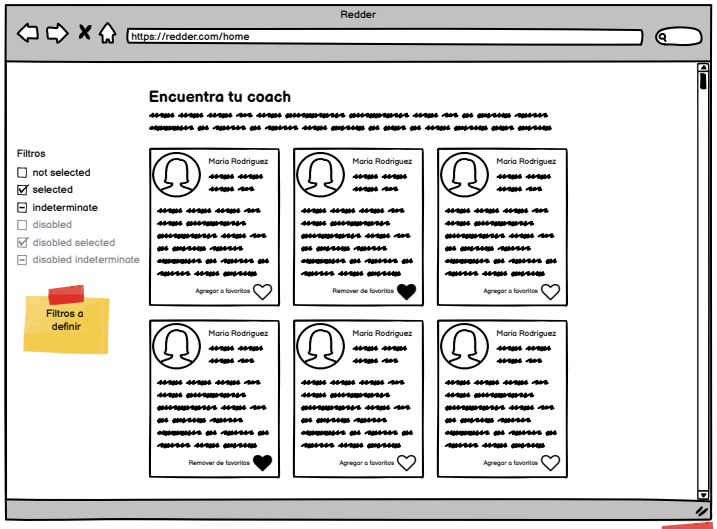


Figura 85: *Maqueta Forgot Password*



Pagina principal luego de iniciar sesión y si no tienes pareja asociada. Se mostrará un feed con los coaches/referentes donde el usuario seleccionara sus favoritos (al menos 3)

Figura 86: Maqueta Home

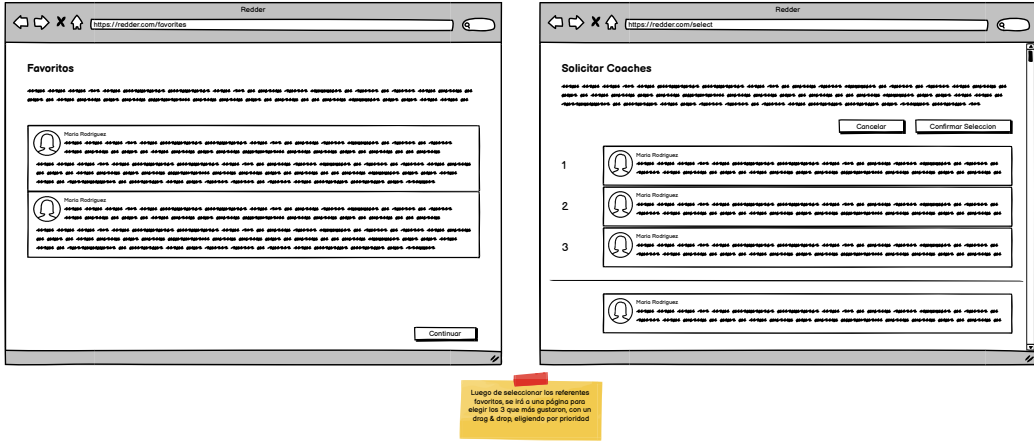
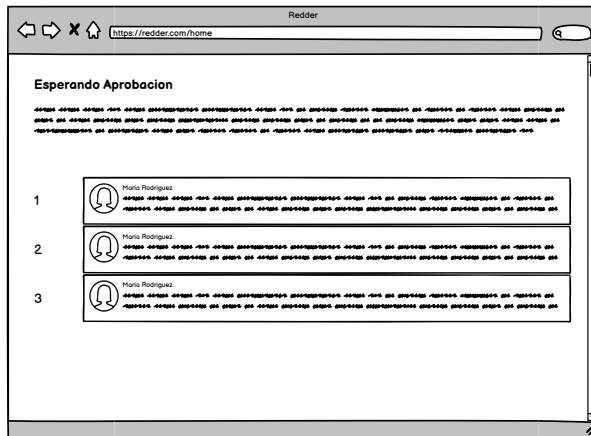
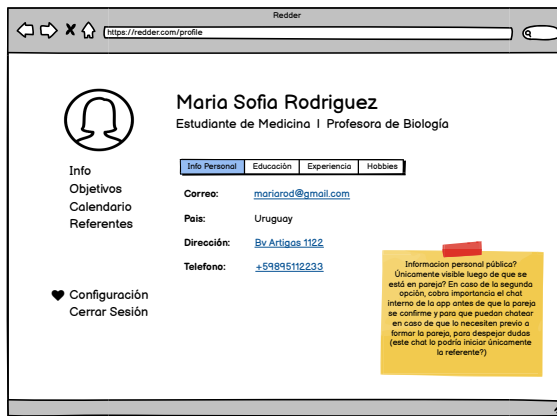


Figura 87: Maqueta Favoritos



Luego de elegir los 3 con más prioridad, se mostrará una página con el estatus del estado de las peticiones de pareja.

Figura 88: Maqueta Aceptar Solicitud



Se quieren diferenciar los perfiles de referentes y referidos? O pueden ser iguales? En caso de diferentes, cuales diferencias se quieren mostrar?

Información personal pública?
Únicamente visible luego de que se está en pareja? En caso de la segunda opción, cobra importancia el chat interno de la app antes de que la pareja se confirme y para que puedan chatear en caso de que lo necesiten previo a formar la pareja, para despejar dudas (este chat lo podría iniciar únicamente la referente?)

Figura 89: Maqueta Perfil Usuario

C. Encuesta realizada a las integrantes del directorio de la red

En este anexo se puede observar la encuesta realizada a las integrantes del directorio de la red, con los correspondientes resultados. Es importante mencionar que la encuesta constaba de cuatro preguntas con puntaje del 1 al 5, donde 1 hace referencia a “Muy en desacuerdo” y el 5 a “Muy de acuerdo”, y una última pregunta opcional para que indiquen comentarios generales si es que así lo deseaban.

Estoy conforme con el proceso de trabajo relativo al equipo de desarrollo de la plataforma

7 responses

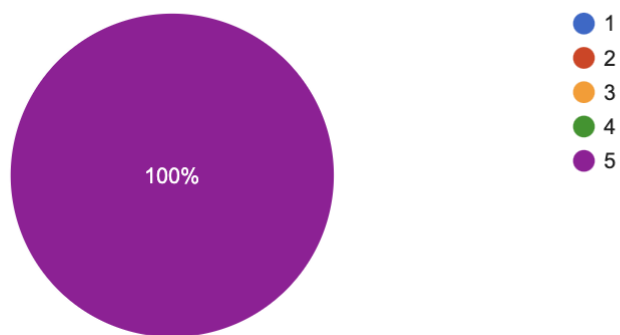


Figura 90: Respuesta pregunta 1

Estoy conforme con el diseño de la plataforma

7 responses

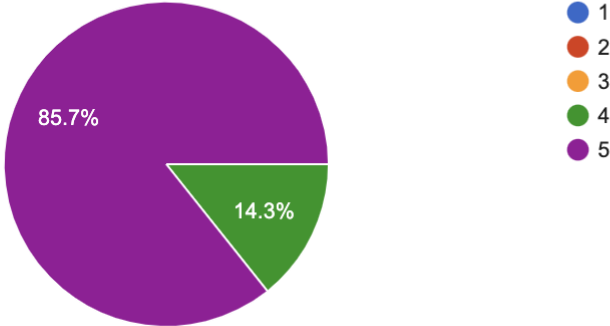


Figura 91: Respuesta pregunta 2

La plataforma resulta sencilla de usar, incluso para personas no familiarizadas con la tecnología

7 responses

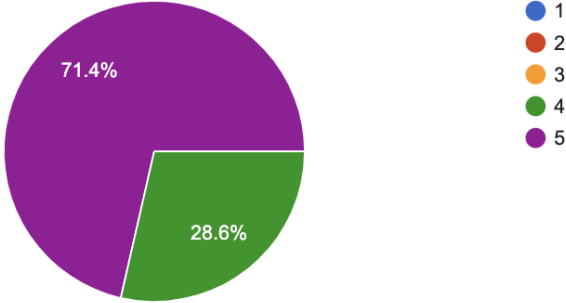


Figura 92: Respuesta pregunta 3

Estoy conforme con el resultado final de la plataforma

7 responses

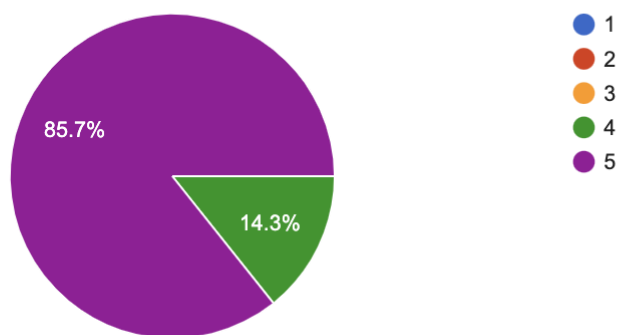


Figura 93: *Respuesta pregunta 4*

Comentarios

2 responses

Excelente herramienta, en pleno uso para el fin que fue diseñada, a la que se le podrán agregar muchas funcionalidades de futuro.

Es una plataforma muy amigable, inclusive para personas que no están familiarizadas con la tecnología. Me resultó muy práctica y sencilla.

Figura 94: *Respuesta pregunta 5*

D. Endpoints Redder API

En esta sección del anexo se pueden observar algunas de las distintas rutas que se definieron para los endpoints que expone la API.

Endpoints relacionados con autenticación:

- POST /auth/register (Registrar usuario)
- POST /auth/finish-register (Terminar registro de usuario)
- GET /auth/google (Autenticación con Google)
- GET /auth/linkedin (Autenticación con LinkedIn)
- GET /auth/verify?verificationtoken=... (Verificar Email)
- POST /auth/verify/send (Reenviar email de verificación)
- POST /auth/recover (Recuperar contraseña)
- POST /auth/reset (Resetear contraseña)
- PUT /auth/password (Cambiar contraseña)
- DELETE /auth/delete (Eliminar cuenta)

Endpoints para manejo de usuarios:

- GET /users/profile (Obtener perfil de usuario)
- PUT /users/profile (Editar info del perfil de usuario)
- GET /users/referrers (Obtener Referentes)
- POST /users/:id/report (Reportar usuario)

Endpoints para manejo de Áreas de Interés:

- POST /expertiseareas (Crear Áreas de Interés)
- PUT /expertiseareas/:id (Editar Áreas de Interés)
- DELETE expertiseareas/:id (Eliminar Áreas de Interés)

Endpoints para manejo de Parejas:

- POST /partnership/:id/meetings (Crear Reunión)

- PUT /partnership/:id/meetings/:id (Actualizar Reunión)
- DELETE /partnership/:id/meetings/:id (Eliminar Reunión)
- POST /partnership/:id/objectives (Crear Objetivo)
- PUT /partnership/:id/objectives/:id (Actualizar Objetivo)
- DELETE /partnership/:id/objectives/:id (Eliminar Objetivo)
- POST /partnership/:id/notes (Crear Nota)
- PUT /partnership/:id/notes/:id (Actualizar Nota)
- DELETE /partnership/:id/notes/:id (Eliminar Nota)

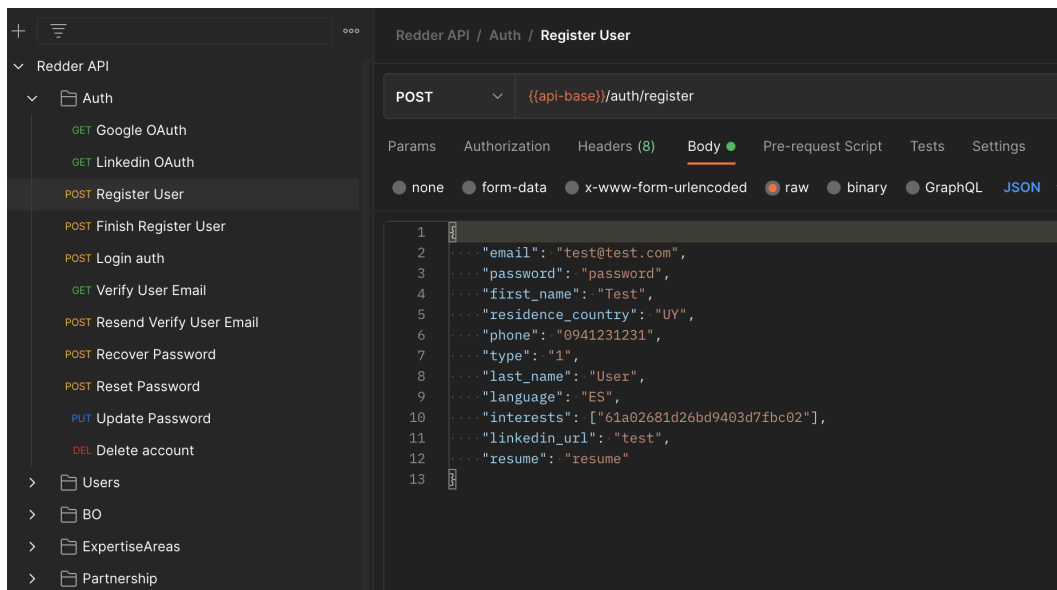


Figura 95: Colección Postman Redder API

Como se puede observar al analizar las urls de los endpoints se puede intuir con solo leer las rutas y el verbo http la función que cumple cada uno.

E. Los principios del Manifiesto Ágil

En este anexo se pueden ver los doce principios que se siguen en el Manifiesto Ágil son: [41]

- “Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.”
- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
- Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- El software funcionando es la medida principal de progreso.
- Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.”

F. Modelo Entidad Relación

En esta sección del anexo se pueden observar con mayor claridad las entidades y sus respectivos atributos, del diagrama Modelo Entidad-Relación (MER) utilizado en la etapa de diseño de la base de datos.

F.1. Usuario

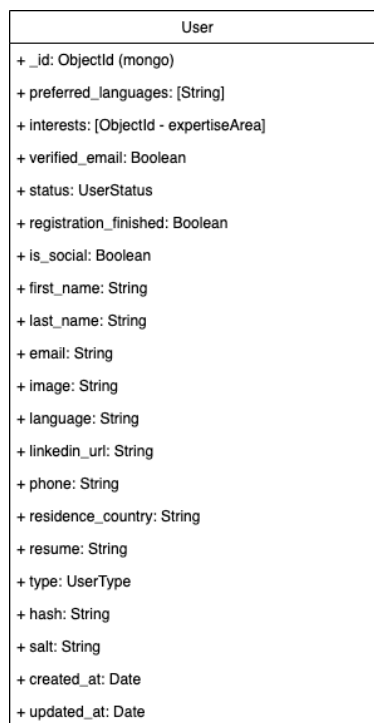


Figura 96: *Entidad de Usuarios*

F.2. Áreas de Interés

Expertise Areas
+ _id: ObjectId (mongo)
+ relates_to: { expertise_area_id: ObjectId (mongo) weight: Number, }
+ user_type: UserType
+ name: String
+ matches_with: { expertise_area_id: ObjectId (mor weight: Number, }
+ created_at: Date
+ updated_at: Date

Figura 97: Entidad de Áreas de Interés

F.3. Tarea Programada

CronJobs
+ _id: ObjectId (mongo)
+ name: String
+ data: { partnership_id: ObjectId (mongo) }
+ priority: Number
+ type: String
+ nextRunAt: Date
+ lastModifiedBy: Date
+ lastRunAt: Date
+ lastFinishedAt: Date
+ lockedAt: Date

Figura 98: Entidad de Tarea Programada

F.4. Pareja

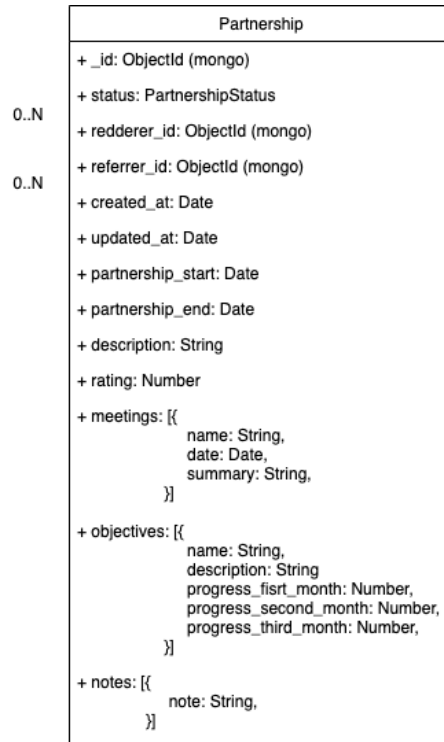


Figura 99: *Entidad de Pareja*

F.5. Reporte

Report
+ _id: ObjectId (mongo)
+ status: ReportStatus
+ reporter_id: ObjectId (r
+ reported_id: ObjectId (
+ reason: String
+ created_at: Date
+ updated_at: Date

Figura 100: *Entidad de Reporte*

G. Pruebas de validación

Plan de Pruebas - REDDER

En este documento se detallan los flujos principales que forman parte de la plataforma REDDER. El objetivo de este documento es que sirva como una guía para realizar las pruebas de regresión para la plataforma REDDER. Los casos que se presentan en este documento son los casos de uso críticos para el funcionamiento de la plataforma.

Guía:

Rechazar solicitud de Referente

1. Realizar un registro como Referente usando email y password
2. Una solicitud de nuevo referente debería llegar al Backoffice (dejándole saber que su registro está en revisión).
3. Desde el backoffice, rechazar la solicitud, con una razón
4. Verificar que el usuario reciba la respuesta a su solicitud a su mail
5. Verificar que el usuario no pueda ingresar a la plataforma.

Aceptar solicitud de Referente

1. Registrarse como Referente
2. Una solicitud de nuevo referente debería llegar al Backoffice (dejándole saber que su registro está en revisión).
3. Desde el backoffice, aceptar la solicitud
4. Verificar que el usuario reciba email notificando que fue aceptado
5. Verifica que este pueda acceder a la plataforma.

Recuperar contraseña

1. Partiendo de un usuario ya registrado
2. Indicar que olvidé mi contraseña, ingresando el email del usuario
3. Verificar que llegue el email a la casilla de correo ingresada
4. Hacer click en el link adjunto
5. Ingresar una nueva contraseña
6. Verificar que el usuario pueda ingresar con la nueva contraseña

Visualizar Referentes según perfil

1. Realizar un registro como Redderer
2. Verificar que el usuario pueda ingresar a la plataforma
3. Verificar que el usuario pueda ver los referentes según sus datos de perfil

Solicitud de emparejamiento

1. Partiendo de un usuario Redderer y un usuario referente ya registrados y que no estén en pareja
2. Ingresar con el usuario redderer
3. Enviar solicitud de emparejamiento al usuario referente mencionado
4. Verificar que el usuario referente puede aceptar o rechazar la solicitud

Listado de solicitudes

1. Partiendo de un usuario Redderer que no esté en pareja
2. Realizar tres solicitudes de referente a cualquier referente
3. Verificar que las solicitudes están visibles
4. Realizar otra solicitud de emparejamiento
5. Verificar que no es posible realizar una nueva solicitud
6. Cancelar una solicitud
7. Verificar que ahora solo hay dos solicitudes en el listado
8. Realizar una nueva solicitud de referente
9. Verificar que ahora existen tres solicitudes de referente

Cancelar una solicitud de emparejamiento

1. Partiendo de un usuario Redderer y un usuario referente ya registrados y que no estén en pareja
2. Ingresar con el usuario redderer
3. Enviar solicitud de emparejamiento al usuario referente mencionado
4. Verificar que el usuario referente puede aceptar o rechazar la solicitud
5. Desde el usuario Redderer, cancelar la solicitud
6. Verificar que el usuario Referente ya no puede visualizar la solicitud

Enviar solicitud de emparejamiento.

1. Partiendo de un usuario Redderer y un usuario referente ya registrados y que no estén en pareja
2. Que sea visible la petición en el perfil del referente.
3. Desde este, cancelar la solicitud.
4. Verificar que le llegue mail al referente indicando el rechazo por parte del referente

Aceptar Solicitud Referente

1. Partiendo de un usuario Redderer y un usuario referente ya registrados y que no estén en pareja
2. Ingresar con el usuario redderer
3. Enviar solicitud de emparejamiento al usuario referente mencionado
4. Verificar que el usuario referente puede aceptar o rechazar la solicitud
5. Desde este, aceptar la solicitud
6. Verificar que el usuario Redderer reciba un correo indicando que la solicitud fue aceptada.
7. Verificar que la pareja quedó creada

8. Verificar que se inició la tarea para terminar la pareja en 3 meses
9. Verificar que las solicitudes enviadas por este Redderer se hayan cancelado.
10. Verificar que este Referente no esté más visible en el feed de los Redderers

Cancelar la pareja por parte del Referente

1. Partiendo de un usuario Redderer y un usuario Referente ya registrados y que estén en pareja entre sí
2. El referente cancela la pareja.
3. Verificar que la pareja quedó cancelada.
4. El Referente queda visible en el feed de emparejamientos.
5. Verificar que el trigger de los 3 meses (de agenda) se haya borrado.

Cancelar la pareja por parte del Redderer

1. Partiendo de un usuario Redderer y un usuario Referente ya registrados y que estén en pareja entre sí
2. El Redderer cancela la pareja
3. Verificar que la pareja quedó cancelada.
4. El Referente queda visible en el feed de emparejamientos.
5. Verificar que el trigger de los 3 meses (de agenda) se haya borrado.

Cancelar pareja por medio del Backoffice.

1. Partiendo de un usuario Redderer y un usuario Referente ya registrados y que estén en pareja entre sí
2. Desde el backoffice se cancela la pareja
3. Verificar que la pareja quedó cancelada.
4. El Referente queda visible en el feed de emparejamientos.
5. Verificar que el trigger de los 3 meses (de agenda) se haya borrado.

Bloquear un usuario.

1. Partiendo de un usuario cualquiera que esté activo
2. Desde backoffice bloquear este usuario
3. Verificar que el usuario ya no puede acceder a la plataforma

Denunciar a un usuario -> Desestimar

1. Partiendo de dos usuario que actualmente estén en pareja
2. Ingresar con uno de los mencionados usuarios
3. Denunciar al usuario
4. Ingresar desde Backoffice
5. Verificar que se puede ver la denuncia realizada
6. Desestimar denuncia
7. Verificar que la denuncia ya no está visible

Denunciar a un usuario -> Aceptar

1. Partiendo de dos usuario que actualmente estén en pareja
2. Ingresar con uno de los mencionados usuarios
3. Denunciar al usuario
4. Ingresar desde Backoffice
5. Verificar que se puede ver la denuncia realizada
6. Aceptar la denuncia
7. Verificar que el usuario denunciado no puede acceder a la plataforma