Facultad de Ingeniería
Universidad Nacional de Mar del Plata

UNMDP - Facultad de Ingeniería

UNIVERSIDAD NACIONAL de MAR DEL PLATA

PUENTE: Plataforma de vinculación tecnológica

Alumno:

- Battista, Franco

Director:

- Genin, Fernando

Co-Director:

- Meschino, Gustavo

Proyecto final para optar al grado de Ingeniero en Informática

Mar del Plata, 29 de febrero de 2024

Facultad de Ingeniería
Universidad Nacional de Mar del Plata

UNMDP - Facultad de Ingeniería

UNIVERSIDAD NACIONAL de MAR DEL PLATA

PUENTE: Plataforma de vinculación tecnológica

Alumno:

- Battista, Franco

Director:

- Genin, Fernando

Co-Director:

- Meschino, Gustavo

Proyecto final para optar al grado de Ingeniero en Informática

Mar del Plata, 29 de febrero de 2024

Agradecimientos

A mis padres por el cariño, soporte y paciencia dados.

A los docentes por la formación y conocimientos recibidos. En especial, a Fernando Genin y Gustavo Meschino, directores de este proyecto, por su buena predisposición al momento de solicitar su participación y ante cualquier inquietud o consulta.

A mis compañeros que conocí a lo largo de la carrera por la colaboración y momentos compartidos.

A la Secretaría de Vinculación Tecnológica por la oportunidad y confianza depositada.

Índice

Resumen

La plataforma Puente, fue concebida en la Secretaría de Vinculación y Transferencia Tecnológica de la Universidad Nacional de Mar del Plata a cargo del Dr. Guillermo Lombera, con el objetivo de centralizar las ofertas tecnológicas disponibles en las distintas unidades académicas de la UNMdP y vincularla con el entramado socioproductivo de Mar del Plata y la región.

El objetivo a lograr de este proyecto es crear un sistema donde los usuarios puedan optimizar la gestión de ofertas y demandas, teniendo así un software único y centralizado para poder acceder a esta información. Además, se busca que agilice el número de vinculaciones entre los actores, gracias a su innovadora Inteligencia Artificial, encargada de encontrar similitudes entre las ofertas y demandas disponibles.

El proyecto fue abordado de forma iterativa e incremental, lo cual facilitó las mejoras del producto a medida que se definían requerimientos y actores del sistema. Dada la complejidad de los requerimientos, se necesitó el desarrollo y entrenamiento de un modelo de Inteligencia Artificial para poder solucionar el problema de la vinculación entre actores, y de la utilización de tecnologías de desarrollo modernas para la creación de un sistema web que fuera robusto, amigable y seguro para los usuarios. Además, para garantizar su funcionamiento se realizaron distintos test de integración para reducir los errores en un alto porcentaje. También se necesitó la utilización de los servidores de la Universidad para poder desplegar la plataforma, lo cual presentó un gran desafío para el desarrollador. Por último, se realizaron capacitaciones para garantizar un buen uso de la plataforma, debido a la dificultad de los usuarios para aprender el uso de la plataforma por su cuenta.

La plataforma se encuentra instalada y funcionando, y se puede acceder desde “https://puente.mdp.edu.ar”. La misma se encuentra alojada en los servidores de la Universidad.

1 - Introducción

En un mundo cada vez más interconectado y dinámico, la innovación y la colaboración se convirtieron en pilares fundamentales para el progreso. En este contexto, la

Universidad Nacional de Mar del Plata decidió trabajar en la creación de una plataforma para la vinculación entre oferentes y demandantes.

Esta herramienta surge de la necesidad de facilitar y potenciar la conexión entre dos esferas claves: la investigación/extensión universitaria y el sector socioproductivo. En un escenario donde la investigación y el desarrollo tecnológico son motores fundamentales del crecimiento, la plataforma PUENTE emerge como un vínculo para promover la colaboración y el intercambio de conocimientos.

El proyecto se basa en dos conceptos claves, que son las ofertas y las demandas. Las primeras, se conocen cómo las ofertas de desarrollo/investigación en la cual los distintos grupos pertenecientes a la Universidad están trabajando. Las segundas, son requerimientos de desarrollo/investigación, confeccionados por usuarios pertenecientes al sector socioproductivo de la ciudad.

En este contexto, el objetivo de este proyecto es lograr crear un sistema donde los usuarios puedan optimizar la gestión de ofertas y demandas, teniendo así un software único y centralizado para poder acceder a esta información.

Es de gran importancia destacar que la centralización de la información en un solo punto se revela como un avance crucial en la plataforma. Anteriormente, la dispersión de datos dificultaba el acceso y la colaboración entre distintos grupos de investigación dentro de la Universidad. Ahora, con Puente, se rompen barreras y se crea un espacio común donde toda la comunidad universitaria puede acceder fácilmente a la información relevante. Esta centralización no sólo simplifica la gestión y consulta de proyectos, sino que también potencia la colaboración y el intercambio de ideas entre las partes. Así, la plataforma no solo cumple el rol de vitrina tecnológica en la UNMdP, sino que también fortalece e impulsa el desarrollo de cualquier tópico dentro de los distintos grupos de investigación.

La plataforma PUENTE permite la visibilidad de las ofertas publicadas, donde cada oferta se presenta de manera clara y accesible, permitiendo a los usuarios explorar fácilmente las oportunidades disponibles. Esta exhibición ordenada y atractiva no solo simplifica el proceso de búsqueda y selección, sino que también aumenta la exposición de los proyectos, atrayendo a un público más amplio de inversores y colaboradores potenciales. En definitiva, la vitrina tecnológica no solo mejora la experiencia de usuario, sino que también potencia el alcance y el impacto de cada oferta, impulsando así la colaboración y la innovación en la plataforma.

La plataforma incluyó una sección dedicada a las líneas de financiamiento provenientes del estado, para que todos los usuarios (anónimos o registrados en la misma) puedan acceder y consultar los términos de cada financiamiento. Esta iniciativa no solo simplifica el proceso de búsqueda y solicitud de financiamiento, sino que también empodera a los usuarios al proporcionarles herramientas para tomar decisiones informadas y estratégicas sobre el futuro de sus proyectos. Además, al centralizar esta información en nuestra plataforma, fomentamos la transparencia y la igualdad de oportunidades, asegurando que tanto los grandes como los pequeños proyectos tengan acceso equitativo a estos recursos financieros.

Además se impulsó el desarrollo de técnicas de inteligencia artificial para la mejora del proceso de vinculación entre oferentes y demandantes. Esta inteligencia artificial no solo optimiza y agiliza el proceso de búsqueda, sino que también enriquece las conexiones al identificar de manera precisa las afinidades y necesidades entre las distintas partes. Este avance tecnológico aumenta la eficiencia de la plataforma, y también abre nuevas posibilidades de colaboración estratégica, impulsando así la innovación y el desarrollo en todos los ámbitos. Además de mejorar la experiencia del usuario, también fortalece la calidad y el impacto de las colaboraciones generadas, consolidando a la plataforma como un motor clave en la promoción del progreso y la transformación.

2 - Análisis del problema

2.1 - Dominio del problema

La Universidad Nacional de Mar del Plata cuenta con un número importante de destacados profesionales dedicados a la investigación y extensión. Como se sabe, este capital humano no es fácil de adquirir, ya que no se encuentra disponible en una tienda.

En la era actual, todo está digitalizado y el mundo no se empeña en buscar este capital de manera activa; más bien, es crucial que esté visible para todos y tenga una mayor exposición para ser reconocido. Esto implica que, si no se desarrolla algún software que respalde este propósito, este capital permanece oculto y sin ser "explotado".

Aunque en muchos casos los desarrollos o investigaciones pueden llevarse a cabo con recursos limitados, la incentivación desde un sector externo siempre contribuye al crecimiento en este ámbito. Obtener recursos adicionales se vuelve esencial, y para lograrlo, es fundamental que estos avances sean visibles mediante alguna vitrina tecnológica virtual, poniendo así sus desarrollos al alcance del sector socioproductivo.

Alcanzar todos estos objetivos es una manera de respaldar y promover el crecimiento de nuestra Universidad, evitando que esos recursos, que podrían buscar conocimiento en otros lugares, se dirijan a instituciones ajenas a nuestra Universidad.

2.2 - Problema a resolver

Como se mencionó de manera resumida, el actual sistema de transferencia y vinculación de la Universidad Nacional de Mar del Plata necesita agilizar su proceso de vinculación entre oferentes y demandantes, para explotar aún más el potencial de los investigadores de la Universidad y aumentar la vinculación con el medio socioproductivo.

En primer lugar, la ausencia de un software dificulta la difusión de ofertas por parte de los docentes/investigadores. Por lo general, estas ofertas de proyecto se publican en redes sociales como Instagram o Twitter (actualmente X), en perfiles con recursos limitados y un número reducido de seguidores. Esto implica que muy pocos se enteran de los desarrollos o investigaciones en curso. Actualmente, la Secretaría de Vinculación Tecnológica de la Universidad ni siquiera tiene conocimiento de la cantidad y características de las ofertas vigentes, ya que no están centralizadas en un punto único de consulta.

Adicionalmente, la presentación de las ofertas no es amigable para el público no especializado. Un ejemplo de la presentación formal de una oferta, anterior a la implementación de Puente, resulta poco atractivo para alguien fuera del ámbito, lo que puede desincentivar la lectura o el interés en el proyecto. Esto se puede apreciar en la *Figura 1.1 y Figura 1.2*.

TRANSFERENCIA y VINCULACIÓN TECNOLÓGICA
subsecretaría UNMDP

INFUSIÓN POR VACÍO PARA FABRICAR PIEZAS DE MATERIAL COMPUESTO

DESCRIPCIÓN

Innovación en el sistema o proceso para la fabricación de piezas de material compuesto basado en el proceso de infusión por vacío con membranas elastoméricas reutilizables.

APLICACIONES

Industrias manufactureras de productos basados en materiales compuestos (resinas epoxi, poliéster, etc. reforzadas con fibras de vidrio, carbono, etc).

VENTAJAS

- Control en tiempo real sobre el flujo de resina durante la impregnación de los tejidos de refuerzo
- Elimina la necesidad de usar materiales descartables propios del proceso de infusión (bolsa de vacío, "peel ply" y "flow media")
- Otorga excelente terminación superficial en ambas caras de la pieza

ESTADO DE DESARROLLO

Se han desarrollado prototipos 100% funcionales a escala de laboratorio (para fabricación de piezas pequeñas con geometrías simples). Se está trabajando en escalar a piezas reales y complejas.

PROPIEDAD INTELECTUAL

Se ha presentado la solicitud de patente y el INPI ha aprobado el examen preliminar técnico de este caso y ha dispuesto la publicación.

INVESTIGADORES

Dr. [redacted]

Figura 1.1. Documento de una oferta previa a Puente

Figura 1.2. Documento de una oferta previa a Puente

Este conjunto de problemas conduce a una baja exposición, lo que impide la generación de contratos de vinculación entre oferentes y demandantes. Esta falta de visibilidad desalienta a los investigadores a participar en estos proyectos o incluso a realizar la carga de ofertas. Por otro lado, los demandantes carecen de una forma eficaz para solicitar desarrollos que se alineen con sus necesidades, lo que resulta en un desperdicio para ambas partes.

En respuesta a todas estas problemáticas, el desarrollo de Puente se vuelve imperativo para contar con una herramienta que posibilite la carga rápida y eficiente, facilitando la vinculación entre oferentes y demandantes. Este proyecto busca movilizar un nicho de mercado estancado, proporcionando una solución integral a los desafíos actuales en la transferencia y vinculación de la Universidad.

2.3 - Flujo de trabajo

Se presenta a continuación el diagrama del Modelo y Notación de Procesos de Negocio (BPMN[1])que ilustra el flujo principal de interacción de un usuario con el sistema, específicamente en la carga de una oferta. Los actores involucrados son: el Oferente, encargado de cargar la oferta; el Demandante, quien recibe notificaciones en caso de coincidencias; el Administrador de la Unidad Oferente, responsable de autorizar (o no) el primer nivel de la oferta; el Administrador de Rectorado, encargado de autorizar o denegar la oferta en el segundo nivel; y finalmente, el Sistema de Matcheo, que realiza la tarea

efectiva de emparejamiento. Se puede observar en el diagrama en la *Figura 2*, que los administradores están juntos en una sola entidad y que el demandante en este caso solo recibe el correo, por lo cual no participa del flujo, y por ende no se lo agregó como entidad en el diagrama.

En este flujo principal, se pueden identificar claramente las distintas etapas por las que pasa una oferta antes de ser publicada en la web. Además, proporciona una representación visual del momento en el que el sistema de matcheo comienza a intervenir. También es evidente que los usuarios se mantienen informados en todo momento sobre el estado de la oferta y en qué fase del proceso se encuentra.

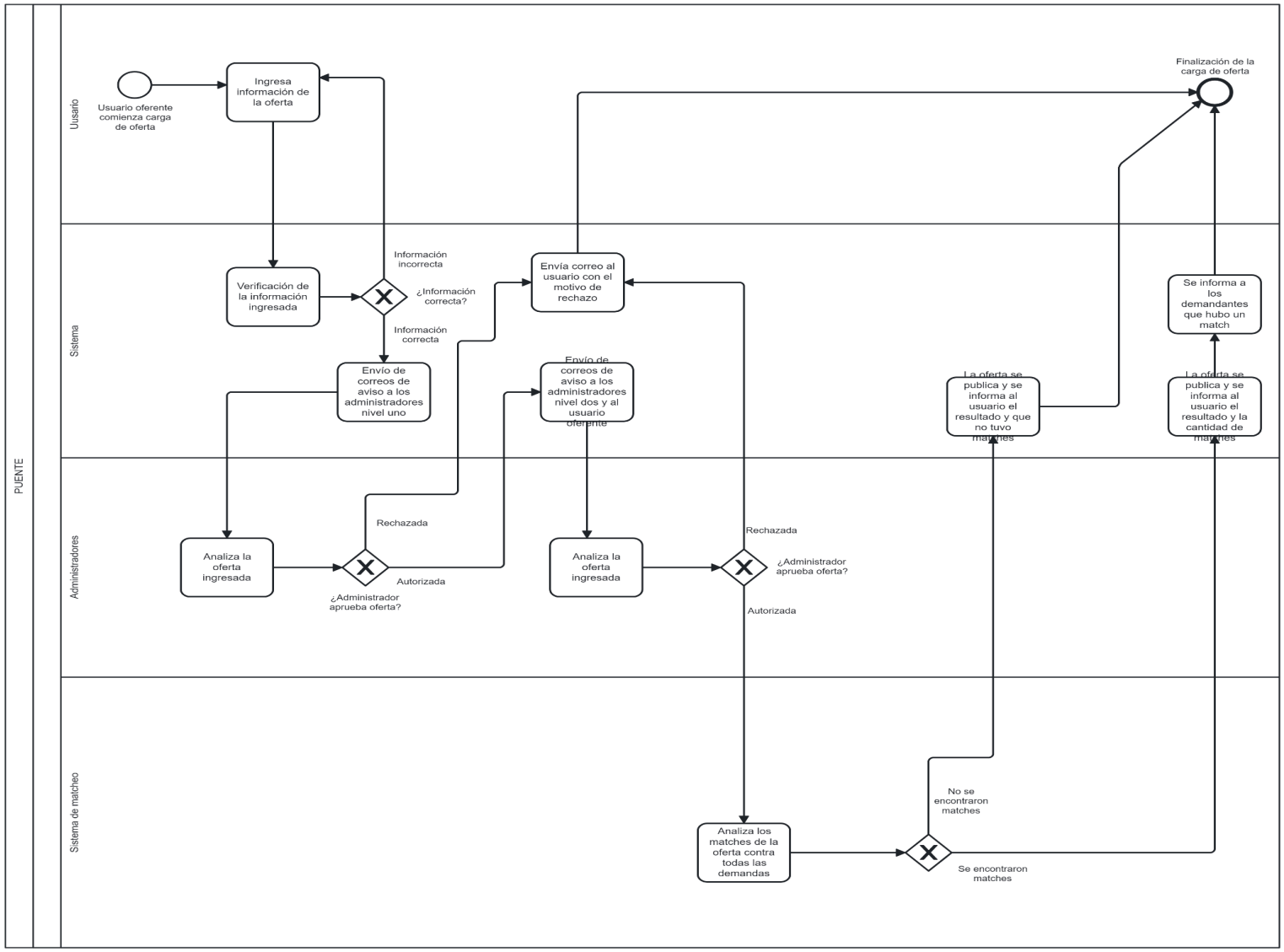

Figura 2. Diagrama de proceso

2.4 - Requerimientos

A continuación se listan todos los requerimientos del sistema, planteados en la etapa de análisis.

Requerimientos funcionales:

- ***RF01:*** El sistema deberá autenticar a los usuarios mediante email y contraseña
- ***RF02:*** El sistema permitirá elegir un tipo de usuario en su creación, siendo este rol demandante y oferente.
- ***RF03:*** El sistema debe permitir asignar roles a otros usuarios mediante un administrador con privilegios superiores.
- ***RF04:*** El sistema permitirá crear, modificar, eliminar ofertas a los usuarios oferentes.
- ***RF05:*** El sistema permitirá crear, modificar, eliminar demandas a los usuarios demandantes.
- ***RF06:*** El sistema mostrará información detallada de una oferta o una demanda en una sección exclusiva.
- ***RF07:*** El sistema deberá contener una sección exclusiva para la exploración de todas las ofertas disponibles en la web.
- ***RF08:*** El sistema deberá contener una sección exclusiva para la exploración de todas las demandas disponibles en la web.
- ***RF09:*** El sistema deberá contar con anillos de autorización, donde los usuarios con roles correspondientes puedan rechazar o autorizar la oferta, antes que sea visible para todo el público. Los anillos serán sucesivos, siendo primer nivel para un tipo de administrador y segundo nivel para otro tipo de administrador.
- ***RF10***: El sistema deberá poder contactar a un responsable de oferta cuando la misma le interese al visitante.
- ***RF11:*** El sistema deberá poder crear, modificar y eliminar líneas de financiamiento para un administrador específico.
- ***RF12:*** El sistema deberá poder mostrar el detalle de una línea de financiamiento en una sección específica.
- ***RF13***: El sistema contendrá usuarios con rol autorizado para crear, modificar y eliminar unidades oferentes, correspondientes a los grupos de investigación.

- ***RF14:*** El sistema permitirá que los usuarios oferentes se podrán asociar a las unidades oferentes que pertenezcan.
- ***RF15:*** El sistema no debe permitir que usuarios que no tengan dominio institucional se registren como oferentes.
- ***RF16:*** El sistema deberá confirmar a los nuevos usuarios mediante email.
- ***RF17:*** El sistema contará con una sección anónima donde se podrán visualizar: ofertas y líneas de financiamiento.
- ***RF18:*** El sistema deberá analizar si una oferta o demanda entrante, tiene similitud contra una cantidad de ofertas o demandas, informando si la misma contiene un match.
- ***RF19:*** El sistema deberá permitir a los oferentes y demandantes ver los matches que tienen sus ofertas o demandas.
- ***RF20:*** El sistema permitirá a los administradores de más alto rol ver los matches de todas las ofertas y demandas.
- ***RF21:*** El sistema deberá informar vía correo las actualizaciones de los estados de las ofertas y matcheos.

Requerimientos no funcionales:

- ***RNF01:*** La aplicación web tendrá una interfaz adaptable para los distintos dispositivos. Es decir, debe ser responsive, tanto para navegadores como para dispositivos móviles.
- ***RNF02:*** La aplicación web debe ser amigable para los usuarios que no cuenten con demasiada experiencia
- ***RNF03:*** La interfaz debe ser atractiva, sobre todo las landing page.
- ***RNF04:*** La interfaz no debe mantener las reglas de diseño de las páginas oficiales de la universidad, que se consideran poco atractivas al usuario.
- ***RNF05:*** La aplicación debe ser escalable para la implementación de módulos posterio***res.***
- ***RNF06:*** La plataforma debe ser segura debido a que maneja información sensible del personal de la Universidad.
- ***RNF07:*** La plataforma debe ser compatible con todos los navegadores.

3 - Proyecto

3.1 - Objetivo General

El objetivo de este proyecto consiste en crear un sistema que simplifique y optimice la gestión de ofertas y demandas. En otras palabras, se desea desarrollar un software de acceso web que permita registrar las ofertas de las Unidades Académicas de la UNMdP y demandas provenientes del sector socioproductivo de Mar del Plata y la zona. Los objetivos más específicos se pueden describir como los siguientes:

- **Disponer de una vitrina tecnológica:** Un apartado exclusivo de las ofertas, donde cualquier usuario puede acceder y observar información acerca de las ofertas disponibles.
- **Utilizar IA para generar vinculaciones entre demandantes y oferentes:** Un sistema de matcheo utilizado para facilitar el contacto entre oferentes y demandantes.
- **Disponer de una gestión de financiamientos:** Donde los administradores podrán gestionar las líneas de financiamiento disponibles.
- **Facilitar la gestión de las ofertas de la Universidad:** Pudiendo así centralizar la carga de ofertas en una sola plataforma, resolviendo el problema del desconocimiento de las ofertas que existen en la actualidad.

3.2 - Alcance inicial

Como funcionalidad principal, el sistema deberá incluir un ABM (Alta, Baja y Modificación) tanto para las ofertas como para las demandas, permitiendo a los usuarios gestionar estas acciones de manera sencilla. Además de la gestión de ofertas/demandas, se proporcionará una interfaz para la administración de los grupos de investigación, denominados internamente como unidades oferentes en la plataforma. En este contexto, los roles correspondientes podrán agregar, modificar o eliminar estas unidades según sea necesario.

A solicitud del usuario, se implementarán dos niveles de autorización para la publicación de ofertas, gestionados por los directores de los grupos de investigación de las

distintas unidades académicas y el rectorado, respectivamente. Posterior a la publicación, se incluirá una sección exclusiva para visualizar en detalle las ofertas, conocida como "vitrina tecnológica".

En términos de seguridad, Puente contará con un sistema de autenticación de usuarios, abordando la protección mediante el uso de tokens, encriptación y otras técnicas para garantizar la seguridad al máximo.

El sistema de matching operará a través de una inteligencia artificial entrenada para evaluar la similitud entre los textos descriptivos de ofertas y demandas.

Finalmente, el usuario destacó la importancia de que la plataforma sea responsiva, por lo que se deberá asegurar su accesibilidad y funcionalidad desde diversos dispositivos móviles a Puente.

3.3 - Entregables del proyecto

Se detalla a continuación los entregables de este proyecto:

- Lista de requerimientos funcionales y no funcionales
- Planilla explicativa sobre la seguridad del software
- Documentación:
    - Diagrama de arquitectura del sistema
    - Diagrama de entidades
    - Documentación de la API
    - Manual de despliegue de la aplicación
    - Documentación contra ataques
    - Video de capacitación de usuarios
- Código fuente del sistema:
    - Backend: Código del servicio de backend codificado en nodeJS, junto a su package.json para conocer los paquetes utilizados
    - Frontend: Código del servicio de backend codificado en nodeJS, junto a su package.json para conocer los paquetes utilizados

- Sistema de matcheo: Código fuente de la API creada en python, y el código para cargar el modelo entrenado de Inteligencia Artificial, junto con los pesos del mismo
- SQL: Código SQL para la creación de la base de datos junto con todas sus tablas, procedimientos, entre otros.

3.4 - Aporte del proyecto

3.4.1 - Beneficiarios

La idea de Puente fue pensada por la Secretaría de Vinculación Tecnológica, precisamente por el secretario Guillermo Lombera, quien desde hace tiempo encontró esta problemática a resolverse, y pudo traducirla en una idea de software que luego fue presentada a Fernando Genin, dando origen al diseño de la plataforma.

Además de ser una mejora para la secretaría de Vinculación Tecnológica, es una gran herramienta para los docentes, investigadores y demandantes del sector empresarial, debido a que hará más fácil la tarea de la vinculación entre las partes.

3.4.2 - Actores del sistema

El impacto de la plataforma se centrará en tres actores principales, que son llamados los actores del sistema. Estos son:

- **Secretaría de Vinculación Tecnológica (Administrador de rectorado):** La plataforma optimizará significativamente el proceso para la Secretaría, que dejará de recibir una gran cantidad de ofertas de proyectos en formato impreso. Ahora, con un simple clic, podrán encontrar, autorizar (o rechazar) y examinar en detalle estas propuestas de manera eficiente. Este actor, será el encargado de autorizar las ofertas en el nivel más alto.
- **Administrador de la Unidad Oferente:** Cada unidad oferente tiene administradores de más bajo nivel que el administrador de rectorado. De la misma forma que el actor Administrador de rectorado, éste autoriza las ofertas en un nivel más bajo, para que luego pasen por el control del administrador superior.

- **Oferentes:** Los oferentes experimentarán una mayor facilidad para gestionar sus proyectos a través de una plataforma amigable. Además, tendrán la oportunidad de presentar sus proyectos al sector socioproductivo, facilitando el contacto con posibles colaboradores. También podrán visualizar las demandas actuales del sector, permitiéndoles orientar sus desarrollos hacia las necesidades del mercado.

- **Demandantes:** La plataforma permitirá a los demandantes visualizar de manera accesible los proyectos disponibles y establecer contacto con ellos con tan solo un clic. Este enfoque fomentará la inversión del sector privado en el capital de la universidad, fortaleciendo así la colaboración entre ambos sectores.

- **Desarrollador:** Gracias a las experiencias recibidas en el desarrollo de Puente, el desarrollador podrá ampliar su historia laboral tanto en tiempo, como en distintas habilidades aprendidas. Lo que en un futuro próximo impactará en las búsquedas laborales.

3.4.3 - Actores del proyecto

Luego, se suman los siguientes actores, que los denominaremos actores del proyecto, ya que no tuvieron impacto como actores del sistema pero sí participaron para que el proyecto pudiera llevarse a cabo. Estos son:

- **Subsecretaría de informática:** Sector que se encarga de gestionar los activos informáticos.
- **Sector de líneas de financiamiento:** Sector que se encarga de actualizar las líneas de financiamiento disponibles.
- **Soporte técnico UNMDP:** Conformado por todos los profesionales del área de cómputos, quienes nos brindan asistencia para las migraciones del sistema.
- **PROCER (Programa de Competitividad de Economías Regionales):** Programa de financiamiento el cual financió Puente. Se tuvieron que generar distintos informes de avances a efectos de demostrar la evolución del proyecto.
- **Equipo directivo:** Integrado por Fernando Genin y Gustavo Meschino.
- **Facultades de la UNMdP + Incubadora de Empresas de la UNMdP:** Conformado por todas las Unidades Académicas (además de INTEMA y CONICET) e incubadora de empresas de la Universidad.

3.4.3 - Análisis FODA

Uno de los aspectos fundamentales de la planeación estratégica lo constituye el análisis situacional, también conocido como análisis FODA (fortalezas, oportunidades, debilidades y amenazas), el cual posibilita la recopilación y uso de datos que permiten conocer el perfil de operación de una empresa en un momento dado, y a partir de ello establecer un diagnóstico objetivo para el diseño e implantación de estrategias tendientes a mejorar la competitividad de una organización. (Universidad Veracruzana, 2009)

Se planteó un análisis FODA del proyecto realizado para obtener un panorama bien definido del desarrollo de la plataforma. Aquí el detalle:

Fortalezas:

- Experiencia del autor en el desarrollo de sistemas web.
- Motivación impulsada por un interés genuino en proyectos de mediana envergadura.
- Contribución significativa al historial laboral del desarrollador.
- Habilidades técnicas avanzadas del autor.

Oportunidades:

- Escasa competencia en el mercado.
- Sector poco explorado, brindando oportunidades de destacarse.
- Posibilidad de alianzas estratégicas con otras instituciones o empresas.

Debilidades:

- Inexperiencia en planificación y desarrollo de plataformas de mediana envergadura.
- Disponibilidad horaria limitada debido a compromisos laborales y académicos.
- Desarrollo realizado por un solo integrante, posiblemente afectando la distribución de tareas, y logrando que el proyecto cueste como uno de mayor envergadura.
- Limitaciones en recursos financieros para la implementación.

Amenazas:

- Resistencia al cambio por parte de usuarios con poca experiencia en software de gestión.

- Posibles retrasos debido a recesos universitarios y baja disponibilidad de referentes funcionales.
- Rápida obsolescencia tecnológica, afectando la sostenibilidad del sistema.

3.4.4 - Análisis de Riesgos

Se realizó un análisis de los riesgos que pudieran generar que el proyecto sea inviable o bien que pudieran afectar su desarrollo, logrando demoras en las entregas o imposibilitando el desarrollo de alguna función. Para obtener valores numéricos, se ponderó la probabilidad de ocurrencia P y su impacto sobre el proyecto Imp en una escala que toma valores desde el 1 al 3. Una opción podría haber sido tomar una escala del 1 al 5, pero se decidió utilizar esta escala (del 1 al 3) debido a que los valores se ajustan perfectamente para nuestro problema. Los valores son, para la probabilidad:

1: Poco probable
2: Probable
3: Muy probable

y para el impacto:

1: Poco impacto
2: Impacto significativo
3: Gran impacto

Luego, estos valores son multiplicados para obtener la ponderación del riesgo analizado. Si el peso final da como resultado un valor mayor a seis, se elaborará un plan de contingencia. Se utilizó seis como umbral superior donde se deben aplicar planes de contingencia, debido a que permite considerar solamente los riesgos que tengan un alto impacto y por lo menos moderada probabilidad de aparición, o que tengan alta probabilidad de aparición y por lo menos moderado impacto.

Riesgo	Descripción	Consecuencia	P	Imp	Peso

R01	Frustración del desarrollador	Atrasos en los tiempos de entrega	1	2	2
R02	Alta complejidad técnica	Atrasos en los tiempos de entrega o imposibilidad de construir algunas funcionalidades	1	2	2
R03	Desconocimiento de las tecnologías	Atrasos en los tiempos de entrega por necesidad de aprender las tecnologías en profundidad	1	2	2
R04	Cambios en los requerimientos	Redefinición de los requerimientos y estimaciones iniciales	2	3	**6**
R05	Estimación de tiempo inadecuada	Disconformidad por parte del demandante y atraso en los tiempos de entrega	3	2	**6**
R06	Aparición de competencia	La plataforma deberá competir y ser mejor que la competencia para captar a los usuarios	1	1	1
R07	Falta de financiamiento	El proyecto se cancela	3	3	**9**
R08	Imposibilidad de coordinar reuniones	Atrasos en los tiempos de entrega	2	3	**6**

R09	Obsolescencia de librerías	Problemas en el mantenimiento de la plataforma y generación de nuevos errores	1	3	3
R10	Resistencia al cambio de los usuarios	Poco uso de la plataforma	2	3	**6**
R11	Abuso de los referentes funcionales	Proyecto crece desmedidamente ante los nuevos requerimientos	1	3	3

3.4.5 - Planes de contingencia

Se han encontrado cinco riesgos con peso mayor o igual a seis, por lo que se elaborará los siguientes planes de contingencia:

- **R04:** Se pactará de antemano el objetivo de la aplicación, con los requerimientos claros de forma que no haya confusiones en los mismos.
- **R05:** Se informará al demandante del error en la estimación inicial y se informará una nueva fecha de entrega final, asumiendo los costos del error.
- **R07:** En caso que el proyecto sea cancelado, se elevará una petición para continuar con el desarrollo para que el autor pueda utilizarlo como proyecto final de carrera.
- **R08:** Se redefinirán las estimaciones en caso de que efectivamente el proyecto se atrase debido al poco compromiso del demandante, advirtiendo este punto desde el comienzo del desarrollo
- **R10:** Mediante las entregas parciales, se validará con usuarios de prueba representantes de grupos quienes nos informarán de su comodidad al usar la aplicación, con el objetivo de lograr hacer los cambios antes de la entrega final

3.4.6 - Análisis de mercado

Antes de iniciar el desarrollo de Puente, llevamos a cabo un exhaustivo análisis de mercado con el propósito de identificar otras plataformas similares. Nuestra intención era capitalizar sus fortalezas y evitar caer en posibles errores previamente cometidos por ellas.

Se especificó el radio de la búsqueda a nivel nacional, ya que se determinó que en este ámbito podrían aparecer aplicaciones con características similares, y para estructuras administrativas similares a la de la Universidad. Luego de hacerlo, se encontró que existe una aplicación web de características muy similares a Puente, que se llama Trampoline[2].

Esta plataforma fue fundada en mayo de 2023 y se presenta como el “Amazon de la propiedad intelectual” (Infobae, 2023) . La misma resulta ser una competencia directa a Puente, donde incluso utilizan terminologías similares como puede ser “match”, “ofertas” y “demandas”. Esta plataforma tiene un alcance internacional, trabajando con Universidades como la UBA, Universidad Hebrea de Jerusalén, entre otras. En líneas generales, se puede corroborar que cumple una función muy similar a la de Puente, pero carece de detalles que se pidieron para el desarrollo de nuestra plataforma. Por ejemplo, Puente cuenta con un módulo de administración de líneas de financiamiento, donde los administradores podrán cargar los financiamientos activos. Además, en Trampoline no se puede determinar administradores para controlar la publicación de ofertas y demandas, como si lo hace Puente. Por lo tanto, utilizar esta plataforma lograría un bajo nivel de control en las ofertas y demandas disponibles..

Para finalizar el analisis, recopilamos información sobre los procesos de vinculación y transferencia tecnológica en otras universidades del país. No obstante, esta información resultó ser de limitada utilidad, ya que los requisitos variaban significativamente respecto a los establecidos por la Secretaría de Vinculación y Transferencia de la Universidad Nacional de Mar del Plata.

Como resultado de este análisis, llegamos a la conclusión de que nuestro proyecto enfrenta una competencia muy similar, por lo que se debería trabajar en los detalles diferenciales de Puente para que no sea reemplazable por esta plataforma ya existente.

3.5 - Estimación inicial

3.5.1 - Planificación

Con los requerimientos y el alcance del proyecto debidamente establecidos, se procedió a realizar la estimación inicial, que abordó la duración y el cronograma de tareas necesarias para el desarrollo del sistema. Aunque los objetivos estaban claramente definidos, la falta de experiencia en la planificación planteó desafíos para seguir rigurosamente el plan propuesto inicialmente. La duración total del proyecto se cumplió según lo previsto, pero no ocurrió lo mismo con la duración específica de la confección de algunos módulos, ya que durante el desarrollo surgieron problemas imprevistos. Esto quiere decir que existieron desvíos, pero no atrasos.

A pesar de las dificultades en la estimación, el integrante del equipo contaba con una comprensión previa de la duración de ciertas tareas, gracias a su experiencia de dos años en el desarrollo frontend. Por lo tanto, las tareas relacionadas con el front fueron estimadas con mayor precisión en comparación con las del backend, la base de datos, el deploy y otras áreas.

Se presentó también el desafío de desconocer los tiempos manejados por los referentes funcionales, lo que llevó a la imposibilidad de coordinar reuniones durante los recesos universitarios, un aspecto que no se había contemplado en la estimación inicial. Además, la periodicidad extendida entre las reuniones generaba etapas de inactividad en el proyecto, resultando en momentos de estancamiento.

La estructura de la estimación inicial comprendía las siguientes etapas:

- Análisis de requerimientos.
- Conformidad del usuario.
- Desarrollo.
- Testing.
- Documentación e implementación.
- Capacitación del usuario.
- Documentación del trabajo final.

Con estas etapas, se procedió a confeccionar un diagrama de Gantt[3] , que es un gráfico de barras y una herramienta que permite gestionar nuestros proyectos que

componen de varias tareas. Es una herramienta gráfica donde se puede visualizar todas las tareas, su secuencia, hitos, entre otros. El diagrama puede apreciarse en la *Figura 3*

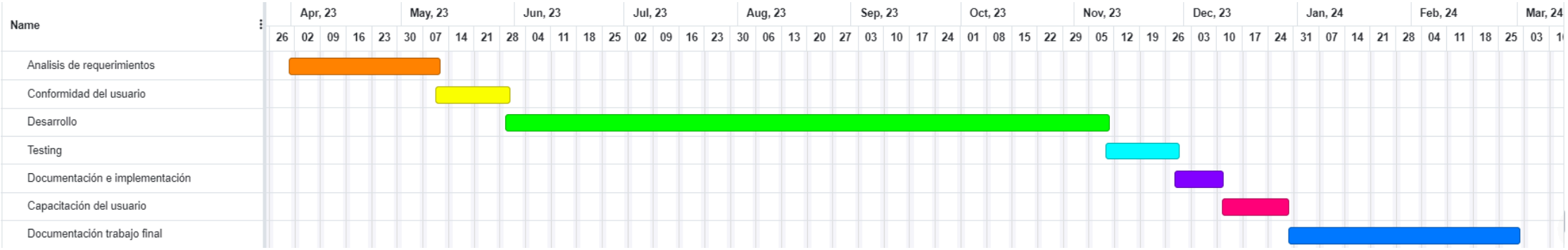

Figura 3. Diagrama de Gantt inicial

Como se observa, la planificación inicial del desarrollo se planteó en etapas de manera lineal, indicando que inicialmente no habría solapamiento entre las diferentes fases. Sin embargo, en la práctica (detalles que se abordarán más adelante en la sección *Planificado vs Ejecutado*), se evidenció que algunas etapas necesariamente se superpusieron entre sí.

3.6 - Metodologías

3.6.1 - Metodología de trabajo

En la fase inicial del proyecto, se eligió la metodología en cascada, que se caracteriza por un enfoque lineal y secuencial en el desarrollo. Como se planteó originalmente, cada fase (análisis, desarrollo y testing) se abordaba de manera consecutiva, y el progreso a la siguiente etapa dependía de la finalización completa de la tarea anterior. Sin embargo, en la práctica, la fase de desarrollo se ejecutó de manera iterativa, marcada por la construcción progresiva de la aplicación módulo por módulo (Digital Talent, 2018).

Este cambio a un enfoque más iterativo se adoptó para mejorar la eficiencia y obtener una mayor aprobación del referente funcional. Después de completar cada módulo, se llevaba a cabo una validación con el cliente, permitiendo ajustes y refinamientos en tiempo real. Las entregas se realizaban mediante deploys parciales en un servidor de prueba, proporcionando a los clientes la oportunidad de realizar pruebas necesarias y analizar los cambios implementados.

La etapa inicial del proyecto estuvo marcada por cierta confusión, ya que era la primera vez que el integrante se enfrentaba a este tipo de enfoque de desarrollo. A pesar de la investigación de diversas metodologías, la elección de una metodología más conservadora, en parte debido a la magnitud del proyecto y la necesidad de seguridad, prevaleció. Aunque no se implementaron metodologías altamente complejas, se hicieron uso de herramientas que simplificaron significativamente la fase de desarrollo, contribuyendo a un proceso más fluido y eficiente.

3.6.2 - Herramientas adicionales utilizadas

Con el objetivo de mantener un orden meticuloso y facilitar la detección de errores, se optó por utilizar Notion[4] como una herramienta integral para el registro de tareas pendientes o completadas durante el desarrollo del proyecto. Notion, gracias a su amplio conjunto de funciones, proporciona a los usuarios una variedad de herramientas que facilitan la organización y estimación de las tareas. En este caso, el desarrollador aprovechó principalmente la capacidad de Notion para dividir los temas de las tareas, marcando claramente cuáles estaban en proceso, finalizadas o pendientes. Aunque Notion puede abordar tareas más complejas, en este contexto se utilizó de manera específica para organizar las tareas y asegurarse de no pasar por alto ninguna.

A pesar de ser un proyecto desarrollado por un único integrante y no requerir la colaboración directa con otro desarrollador, se implementó GIT[5] para el control de versiones de Puente. Este uso de GIT tuvo principalmente la función de servir como un respaldo del código, aprovechando la plataforma GitHub[6] para almacenar repositorios de git de manera segura. Se utilizaron dos ramas o branches para gestionar el código: una rama denominada "main" para el código deployado y otra llamada "dev" para el código en desarrollo. Este enfoque de control de versiones contribuyó a mantener la estabilidad del código y facilitó la gestión de cambios en el proyecto.

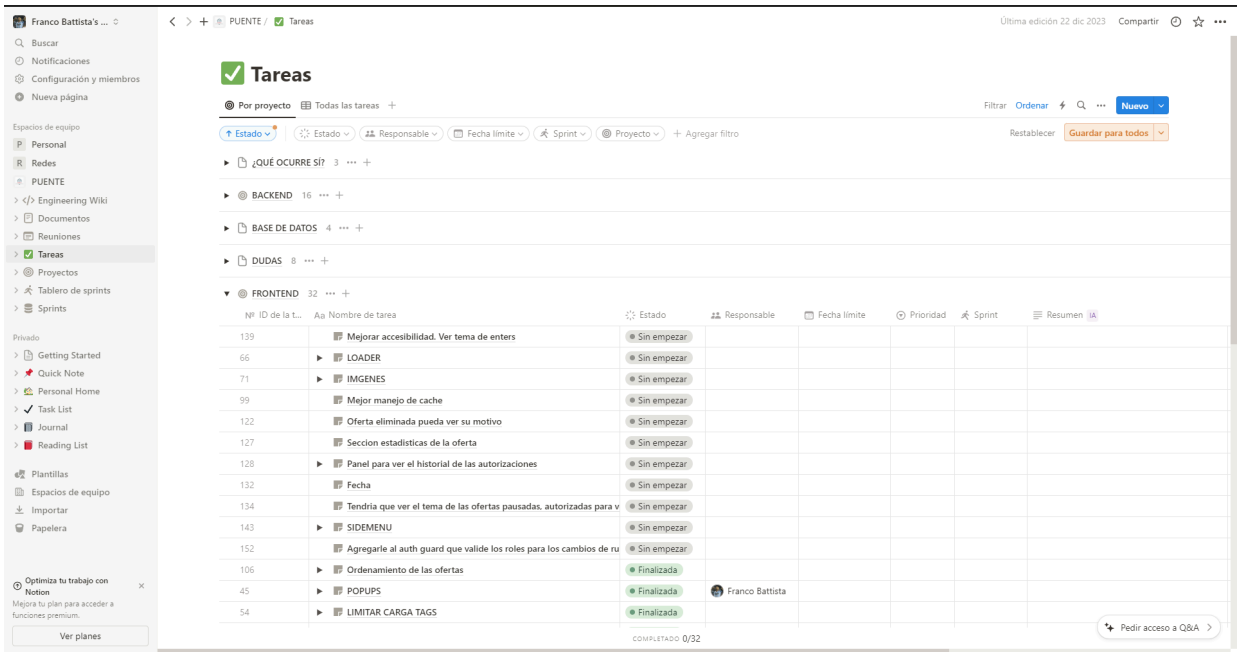

Figura 4. Panel de tareas de Notion

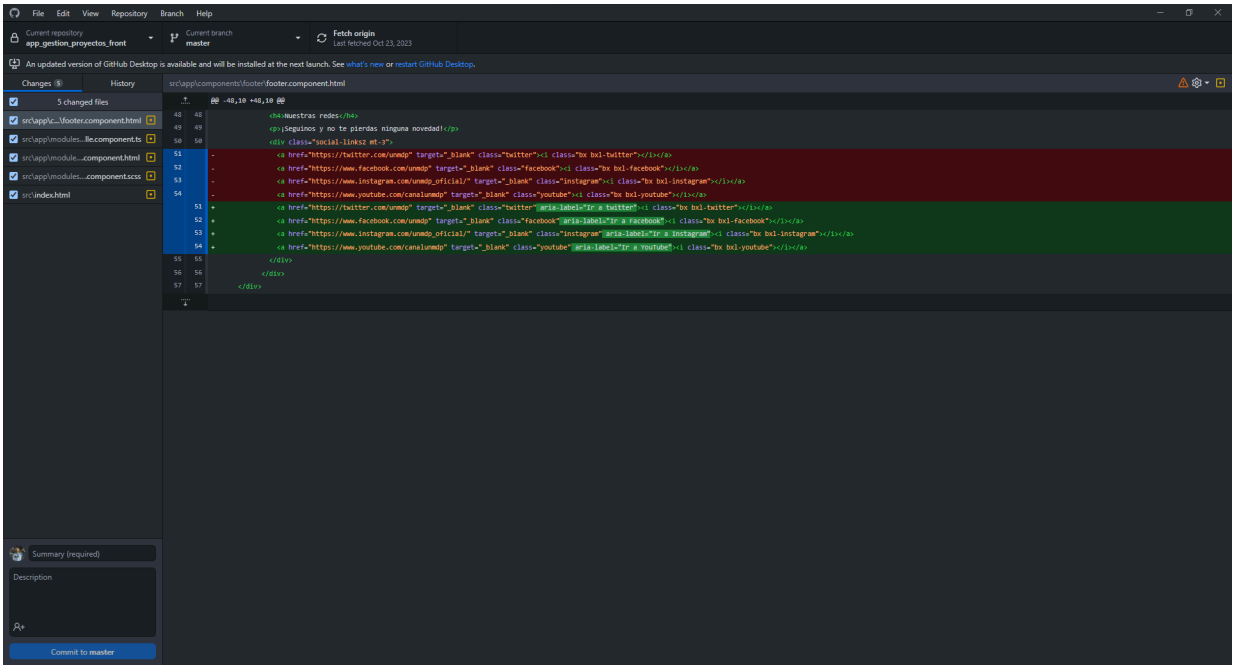

Figura 5. Repositorio del front end con GitHub desktop

3.6.3 - Comunicación

Dado que el grupo estaba compuesto por un único integrante, no era necesario realizar llamadas diarias para que el proyecto avanzara día a día. A pesar de esto, se llevaron a cabo frecuentes reuniones mediante plataformas como "Google Meet" o "Jitsi" con el tutor, Fernando Genin. Estas reuniones permitieron informar sobre los avances desde la última sesión y validarlos desde la perspectiva del tutor. Esta práctica resultó invaluable para identificar posibles errores o áreas de mejora en el desarrollo hasta ese momento.

Las reuniones con el referente funcional y las capacitaciones con los usuarios, en cambio, se llevaron a cabo de manera presencial. Por lo general, estas reuniones se realizaban en la oficina de Guillermo Lombera, ubicada en el edificio de rectorado de la Universidad, situado en las calles "25 de Mayo" y "San Luis". La elección de reuniones presenciales contribuyó a una comunicación más directa y efectiva con el referente funcional, así como a facilitar la interacción con los usuarios durante las sesiones de capacitación. Se explicará más sobre las capacitaciones y sus resultados en la sección “Capacitaciones”.

4- Diseño del sistema

En esta sección, se detallarán en profundidad todos los aspectos técnicos de la plataforma.

4.1 - Arquitectura

Con el fin de proporcionar un entendimiento detallado, se presenta una imagen que ilustra la arquitectura general del sistema. Para Puente, se ha elegido un diseño de arquitectura cliente-servidor[7], en el que los clientes son los usuarios web y el backend adopta una estructura monolítica. La elección de una estructura monolítica en lugar de una orientada a microservicios[8] se basa en el hecho de que solo se dispone del servidor de la Universidad para alojar el backend. Alojar todos los microservicios en este servidor podría generar una sobrecarga significativa de comunicación entre ellos. Además, desarrollar esta arquitectura implicaría un esfuerzo considerable sin beneficios sustanciales. Por ejemplo, si el servidor de la Universidad fallara, todos los microservicios dejarían de estar disponibles, eliminando así la ventaja clave de los microservicios: la disponibilidad continua cuando uno de ellos falla.

A pesar de lo mencionado anteriormente, el sistema cuenta con un servicio adicional, el sistema de matching. En este caso, Puente opera con un backend monolítico que alberga todas las funcionalidades del sistema, excepto el sistema de matcheo. Este último funciona en un puerto diferente y se comunica exclusivamente con el backend principal. La implementación de esta estructura no sigue la lógica de adoptar microservicios, sino que se ha llevado a cabo así debido a que el sistema de matcheo está construido con Python[9]. Se optó por levantar un API exclusivo para este sistema, permitiendo su funcionamiento de manera independiente al monolito principal. Estos detalles se abordarán con mayor profundidad en la siguiente sección.

La plataforma opera como una SPA[10] (Single Page Application) en el lado del cliente, lo que implica que solo se necesita un único archivo HTML[11] del servidor. A través del empleo de un enrutador de JavaScript[12] , se gestionan los componentes que se muestran y ocultan en la pantalla del usuario al cambiar de ruta. En otras palabras, cada vez que hay un cambio de ruta, el cliente no realiza una solicitud GET para obtener archivos renderizados. Esta elección se basa en la decisión de NO utilizar técnicas de SSR[13] (server-side rendering, renderizado del lado del servidor), lo que significa que todo el renderizado ocurre en el lado del cliente. En este enfoque, la API[14] solo devuelve recursos e información, excluyendo la entrega de archivos estáticos como HTML y CSS[15].

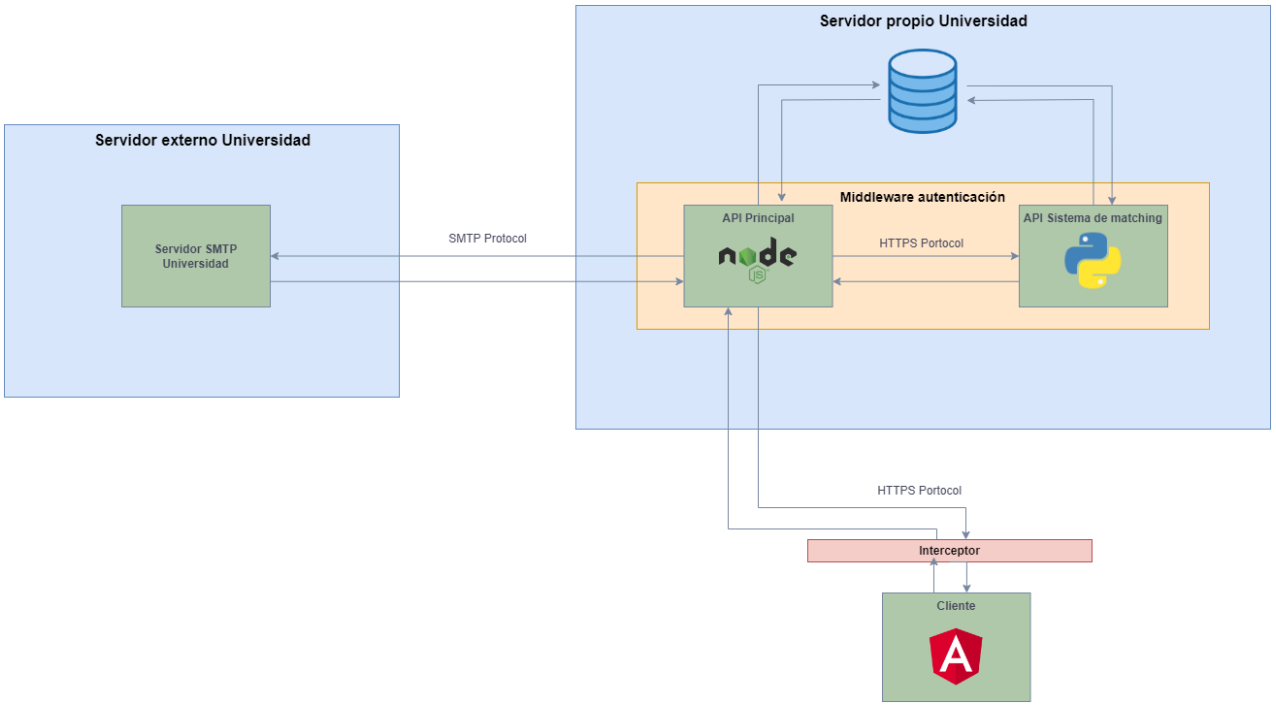

Figura 6. Arquitectura general del sistema

4.2 - Backend

4.2.1 - Tecnologías

Antes de abordar el desarrollo del backend de Puente, se llevó a cabo un análisis detallado para seleccionar las tecnologías que mejor se adecuarán a la solución. Se consideraron varios puntos, como las tecnologías permitidas en el servidor de la Universidad, ya que utilizar un lenguaje no aceptado implicaría una pérdida considerable de tiempo en migrar a otra tecnología. Además, se tuvo en cuenta la experiencia del desarrollador en diversos lenguajes de programación. Dado que este proyecto representaba la primera vez que el autor afrontaba un desarrollo de semejante magnitud, se adoptó una postura conservadora, evitando aprender un nuevo lenguaje durante el transcurso del desarrollo. Por último, se consideró la disponibilidad de documentación en línea y la existencia de una comunidad activa para el lenguaje seleccionado, con el objetivo de contar con soluciones rápidas a posibles problemas durante el desarrollo.

En función de estos criterios, se optó por utilizar NodeJS[16] en el lado del servidor, específicamente en la versión 16.16.0, que era estable en el momento de iniciar el desarrollo. NodeJS, basado en JavaScript, es uno de los lenguajes más utilizados para el desarrollo de backend y cuenta con un amplio soporte para todos los sistemas operativos.

Para la implementación de la plataforma, no se utilizaron únicamente paquetes nativos de NodeJS, sino que se incorporaron diversos módulos de terceros para facilitar el desarrollo. Esta decisión se tomó debido a que la plataforma requería realizar varias tareas que se beneficiaban del uso de módulos externos, permitiendo así un desarrollo más ágil y con menor propensión a errores.

Por último, se menciona que la ruta base para el acceso al backend es: https://puente.mdp.edu.ar/**api/**endpoint.

4.2.2 - Paquetes de terceros principales

En la *Figura 7* se presentan todos los paquetes utilizados, con la imagen del archivo "package.json", que aglutina todos los paquetes con sus respectivas versiones. Este archivo facilita la instalación del proyecto para los usuarios mediante un gestor de paquetes como NPM[17] (Node Package Manager), que permite instalar todos los paquetes mencionados con tan solo un comando.

En el desarrollo de la API se empleó principalmente el paquete llamado "Express[18]". Esta elección se hizo en lugar de utilizar el módulo nativo "http", ya que Express proporciona numerosas facilidades para la creación de distintos endpoints y la modularización de la aplicación. Además, es ampliamente utilizado, lo que garantiza un sólido respaldo y soporte comunitario.

En la sección *'Seguridad'* que detallará los aspectos específicos, se explicará el uso de los paquetes "bcryptjs" para la encriptación de la información que se guarda en la base de datos y "jsonwebtoken" para verificar las sesiones de los usuarios.

Para la conexión con la base de datos MySQL[19], se optó por el módulo "mysql2[20]", que ofrece ventajas significativas contra las inyecciones SQL y otros tipos de ataques. Además, es un paquete bien probado y cuenta con un soporte constante.

Finalmente, para la gestión del envío de correos electrónicos, que también se detallará en la sección *SMTP*, se utilizó el módulo "nodemailer[21]". Este módulo facilita el envío de correos electrónicos con solo las credenciales de un servidor SMTP[22] (Simple Mail Transfer Protocol).

Figura 7. Contenido del archivo 'package.json'

4.2.3 - Patrones de diseño

Para la creación de la estructura del backend se eligió una arquitectura que incorpora capas y módulos, con una clara separación de responsabilidades entre ellos. Esta estructura se diseñó con el propósito de gestionar de manera eficiente el acceso a la base de datos, la resolución de las solicitudes y la generación de respuestas para las peticiones realizadas. Además, se implementaron diversos middlewares para simplificar las respuestas al cliente, manejar los errores de manera efectiva y tener un mayor control sobre las peticiones de los usuarios. En detalle, la arquitectura de capas se distribuyó de la siguiente manera:

- **Capa de Acceso a la Base de Datos**: Encargada de gestionar las operaciones de lectura y escritura en la base de datos. Este nivel se comunica con la capa de lógica de negocio y maneja las consultas SQL mediante el uso del módulo "mysql2".

- **Capa de Lógica de Negocio:** Contiene la lógica central de la aplicación y se comunica con la capa de acceso a la base de datos para realizar operaciones sobre los datos. Aquí se encuentran los módulos que definen las reglas y procesos específicos de la aplicación.

- **Capa de Controladores:** Responsable de recibir las peticiones HTTP[23], invocar la lógica de negocio correspondiente y enviar las respuestas adecuadas. Esta capa se comunica con los controladores específicos de cada módulo.

- **Capa de Rutas:** Gestiona las rutas de la API y direcciona las peticiones entrantes a los controladores correspondientes. Cada módulo tiene su propio conjunto de rutas, lo que facilita la modularización y el mantenimiento del código.

  Además de la arquitectura por capas, se incorporaron middlewares para:

- **Respuestas al Cliente:** Simplifican la construcción y envío de respuestas al cliente, gestionando aspectos como los códigos de estado HTTP y los formatos de respuesta.

- **Manejo de Errores:** Facilitan la detección y manejo de errores, proporcionando respuestas claras y adecuadas en caso de fallos en la ejecución de las operaciones.

Esta estructura modular y estratificada facilita la comprensión del código, mejora la mantenibilidad y permite un desarrollo más eficiente y controlado de la aplicación backend.

4.2.4 - Módulos

La aplicación está estructurada mediante diversos módulos con el objetivo de lograr una alta escalabilidad. Aunque estos módulos no están distribuidos en distintos

microservicios, se mantiene una estructura donde cada módulo se ocupa de una parte específica, facilitando la detección de errores y el posterior crecimiento del sistema.

Cada uno de estos módulos contiene funciones y maneja distintos endpoints según lo requerido. A continuación, se detallan los módulos y su funcionalidad:

- **Anonymous:** Encargado de tareas que no requieren autenticación, es decir, para usuarios no logueados. Ruta base: /api/anonymous.
- **Demandas:** Contiene los endpoints relacionados con las demandas. Ruta base: /api/demanda.
- **Líneas de Financiamiento:** Contiene los endpoints relacionados con las líneas de financiamiento. Ruta base: /api/financiamiento.
- **Mail sender:** Encargado de todo lo relacionado con el envío de correos electrónicos. No posee endpoints, sino métodos llamados en diversas ocasiones.
- **Matching System:** Módulo que comunica con el servicio del sistema de matcheo. Tiene endpoints y también métodos que preparan el matcheo y analizan la respuesta. Ruta base: /api/matching.
- **Unidades Oferentes:** Contiene los endpoints relacionados con las unidades oferentes, nacts y relacionados. Ruta base: /api/nact.
- **Ofertas:** Contiene todos los endpoints relacionados con las ofertas. Ruta base: /api/oferta.
- **Admin General:** Contiene todos los endpoints que puede realizar un usuario con rol de administrador general, es decir, el admin con mayor autoridad del sistema. Ruta base: /api/admgeneral.
- **Usuarios:** Contiene los endpoints relacionados con los usuarios, como el inicio de sesión y registro. Ruta base: /api/user.

Además de los módulos, existen varios archivos que contienen configuraciones o métodos particulares, como:

- **Helper:** Archivo que contiene métodos generales utilizados en toda la aplicación. Es importado en casi todos los módulos.
- **Swagger:** Archivo de configuración del paquete "Swagger[24]", utilizado para la documentación de la API.

- **Logger:** Archivo de configuración del sistema de archivos de LOG para tener una bitácora de errores más detallada.
- **Database:** Contiene todos los métodos de acceso a la base de datos, importando el módulo "mysql2" y manejando las conexiones con la base de datos.
- **Config:** Archivo donde se recopila toda la configuración del sistema, obteniendo todas las variables de entorno a utilizar.
- **MulterConfig:** Configuración del módulo "Multer[25]", utilizado para el manejo de imágenes.

4.2.5 - Manejo de errores

Como se mencionó anteriormente, se desarrolló un middleware específico para gestionar de manera eficiente los errores y proporcionar respuestas adecuadas al cliente en caso de fallos. Utilizando el módulo Express, se aprovechó la capacidad de terminar una petición en un middleware de salida predefinido cada vez que se desee. De esta manera, cuando surge un error, se redirige a un middleware de salida de error, donde se elabora la respuesta con el correspondiente aviso de error.

En la construcción de este middleware, cada vez que se produce un error, se asigna un código y un mensaje de error. En el backend, existe una tabla que asigna a cada código un mensaje específico y más detallado. Esta tabla de errores permite proporcionar respuestas más informativas y comprensibles para el usuario. Por ejemplo, si el error proviene de la base de datos, se asigna una etiqueta llamada "SQL" y un código de error SQL específico, permitiendo que la tabla de errores contenga un mensaje preciso para ese tipo de error relacionado con la conexión a la base de datos.

4.3 - Frontend

4.3.1 - Tecnologías

En cuanto al frontend, cualquier tecnología seleccionada cumpliría los mismos requisitos técnicos, ya que, en el momento de la compilación, todo se traduce en archivos

HTML, CSS y JavaScript. Por lo tanto, el factor principal que influyó en la decisión de la tecnología a utilizar fue la experiencia del desarrollador en esa tecnología. En consecuencia, se optó por Angular[26] con Typescript[27].

Angular, un framework de JavaScript orientado a componentes desarrollado por Google, facilita el desarrollo de una SPA (Single Page Application) de manera más sencilla, gracias a las diversas herramientas que proporciona. Su enfoque basado en componentes beneficia la escalabilidad de la aplicación, que se construye en formato de "árbol" mediante la inserción de componentes unos dentro de otros. Por estas características, Angular se considera ideal para el desarrollo eficiente de una SPA.

Por otro lado, Typescript es un superset de JavaScript que agrega tipado estático al lenguaje. Esta adición resulta beneficiosa para generar código más ordenado y facilita la detección de errores futuros durante el desarrollo.

4.3.2 - Paquetes de terceros principales

Durante el proceso de desarrollo del frontend con Angular, se aprovechó la capacidad del framework para integrar paquetes de terceros, lo que contribuyó significativamente a simplificar y mejorar la eficiencia del desarrollo. La versatilidad de Angular para incorporar bibliotecas externas permitió extender las funcionalidades y capacidades del framework base. Sin embargo, se adoptó una estrategia selectiva al elegir paquetes de terceros, priorizando aquellos que aportaran un valor concreto y necesario para la solución planteada.

A pesar de la flexibilidad ofrecida por Angular para integrar paquetes de terceros, se optó por una aproximación más moderada en el uso de estas adiciones. En lugar de depender en gran medida de paquetes externos, se enfocó en la utilización de frameworks específicos para el diseño de interfaces, maximizando así la coherencia y la eficiencia del desarrollo. La elección de paquetes se realizó cuidadosamente para evitar una dependencia excesiva y garantizar la estabilidad de la solución.

Entre los frameworks más significativos utilizados para el diseño de interfaces se encuentran:

- **Bootstrap**[28]: Este framework de código abierto se destaca por proporcionar herramientas y estilos predefinidos que facilitan la creación de interfaces y diseños responsivos. La aceptación generalizada y la amplia comunidad de usuarios respaldan su elección como una opción robusta para establecer la apariencia inicial del sistema.

- **MaterialUI**[29]: Desarrollado por Google, MaterialUI se basa en los principios de diseño de interfaces y componentes de Material Design, comúnmente asociados con dispositivos Android. La integración de MaterialUI aporta coherencia visual y componentes modernos que contribuyen a una experiencia de usuario atractiva y actualizada.

- **Tailwind CSS**[30]**:** Este framework se centra en la codificación eficiente de estilos, proporcionando una mayor simplicidad a través de clases predefinidas que abrevian los estilos. La utilización de Tailwind CSS agiliza el desarrollo y la personalización de estilos, mejorando la legibilidad y mantenimiento del código en comparación con enfoques más tradicionales.

4.3.3 - Patrones de diseño

El framework Angular se basa en un diseño orientado a componentes. Esto significa que puedes crear componentes reutilizables para diferentes partes del código, como rutas o botones. Esta capacidad de reutilización y modularidad facilita el desarrollo y mantenimiento del código, ya que los componentes pueden ser fácilmente integrados en distintas secciones de la aplicación. La estructura resultante sigue un modelo de árbol, donde los componentes se inyectan unos dentro de otros de manera jerárquica, conformando así un árbol de componentes. A cada uno de estos módulos generados, se le generó un archivo de módulo específico donde pueden utilizar los paquetes que necesiten, generando así un código mucho más limpio y escalable.

En adición a este enfoque de diseño basado en componentes, Angular también implementa el patrón de diseño Modelo-Vista-ViewModel (MVVM). En el contexto de Angular, cada componente actúa como una combinación de Vista y ViewModel. La Vista representa la interfaz de usuario, mientras que el ViewModel maneja la presentación de datos y las interacciones del usuario. Esta combinación de patrones proporciona una

arquitectura sólida y modular que facilita el desarrollo de aplicaciones web eficientes y mantenibles.

4.3.4 - Módulos

El desarrollo del frontend se estructura en módulos independientes que han sido creados por el desarrollador para garantizar la independencia entre ellos. Cada uno de estos módulos opera de forma autónoma y hace uso de diferentes paquetes para evitar la redundancia de recursos en módulos no relacionados. Cada módulo representa una sección independiente de la aplicación que puede funcionar sin depender de otros.

Los módulos incluyen:

- **Componentes:** Contiene todos los elementos reutilizables, como botones y otros componentes que se emplearán en diferentes partes de la aplicación.
- **Anonymous:** Engloba las rutas y componentes utilizados en la parte anónima de la aplicación, brindando la experiencia para usuarios no autenticados.
- **General:** Aborda aspectos generales que no están específicamente asignados a otras secciones o módulos definidos.
- **Auth:** Se encarga de la autenticación y gestión de sesiones de usuarios.
- **Gestión de demandas:** Gestiona las funcionalidades relacionadas con las demandas dentro del sistema.
- **Gestión de ofertas:** Se encarga de las operaciones vinculadas a la gestión de ofertas presentes en la plataforma.
- **Gestión de usuarios:** Administra las operaciones y funcionalidades relacionadas con la gestión de usuarios en la aplicación.
- **Home Landing Page:** Contiene los elementos y componentes específicos de la página de inicio.
- **Mi Usuario:** Enfocado en la gestión de información y ajustes del perfil de usuario.
- **Gestión Nact:** Se ocupa de las acciones y procesos relacionados con las Nact (Unidades Ofertantes).
- **Gestión de Líneas de Financiamiento:** Gestiona las operaciones vinculadas con las distintas líneas de financiamiento disponibles en la plataforma.

4.3.5 - Manejo de errores

En este caso, el manejo de errores se realizó en colaboración con el backend. Cuando se producía algún error, como una falla en la autenticación o un error en el servidor, la respuesta desde el backend contenía una ruta a la cual se redireccionará al usuario. Esta gestión se centraliza desde un interceptor ubicado en el cliente, el cual maneja las salidas y respuestas, y analiza qué acciones debe realizar en cada caso. Además, cada vez que se genera una respuesta con error, se genera un pop-up que informa al usuario sobre la razón del error.

4.4 - Componente innovador

Aunque este proyecto representa un desarrollo de considerable envergadura, su naturaleza principal es la de una transformación digital. Sin embargo, la plataforma incorpora un componente distintivo que, hasta la fecha de investigación, no se ha encontrado en el mercado. Este componente es el sistema de matcheo. En términos concisos, el sistema de matcheo toma una descripción de una oferta o una demanda, analiza su similitud con las demandas u ofertas disponibles en ese momento y genera diversos coeficientes para cada demanda u oferta. Finalmente, establece un "match" si el coeficiente supera un umbral predefinido. Todos estos aspectos se explorarán con más detalle en la sección dedicada al "*Sistema de Matcheo*".

4.5 - Sistema de matcheo

4.5.1 - Definiciones

Las computadoras son capaces de realizar cálculos y operaciones a gran velocidad, lo cual las hace extremadamente eficientes para resolver tareas matemáticas. Sin embargo, durante mucho tiempo han enfrentado una limitación significativa: su dificultad para imitar y emplear la razón de la misma forma que los seres humanos. Un sistema informático maneja información en forma de unos y ceros, conocido como lenguaje binario. En contraste, los seres humanos nos comunicamos mediante el lenguaje natural. Esta diferencia ha dificultado que las computadoras comprendieran este tipo de lenguaje, lo que ha limitado su capacidad para realizar ciertas tareas, como el análisis de texto. En este contexto surge la inteligencia artificial[31], que *“busca que las computadoras hagan lo mismo que nuestro cerebro"* (Boden, 2018, p. 1).

En este contexto, se han generado una gran cantidad de oportunidades de desarrollo que antes no existían, permitiendo a las computadoras procesar nuestro lenguaje y darnos soluciones que en el pasado no eran posibles. Dado que las características del problema requieren que un sistema analice nuestro lenguaje natural, dándonos una respuesta en función de la similitud de dos textos, se optó por utilizar un modelo basado en inteligencia artificial, descrito en esta sección.

4.5.2 - Tecnologías

El sistema de matcheo claramente tenía que implementarse mediante un tipo de inteligencia artificial que pudiera encontrar el match entre una oferta y demanda. La tendencia actual que se aprecia es que todo lo relacionado con inteligencia artificial se desarrolla en Python, tanto entrenamiento como uso de una red neuronal. La comunidad de Python respecto a la inteligencia artificial es enorme, junto con mucho soporte a todos los paquetes de terceros para el desarrollo de la misma. Es por eso que se decidió utilizar Python 3.11.1 para el desarrollo del mismo.

4.5.3 - Paquetes de terceros principales

Para llevar a cabo esta implementación, se utilizaron varios paquetes esenciales que proporcionan funcionalidades clave al sistema Puente. A continuación, se describen brevemente cada uno de ellos:

FastAPI[32]: FastAPI fue fundamental para la creación de la API en Python de manera intuitiva y sencilla. Su estructura es similar a la de Express en Node.js, ofreciendo todas las herramientas necesarias para establecer endpoints de manera eficiente. Facilita la creación de servicios web y garantiza un manejo eficaz de las solicitudes y respuestas.

Sentence Transformers[33]**:** Este paquete resultó esencial para descargar e implementar el modelo preentrenado utilizado en el sistema. Internamente, utiliza Torch y proporciona los métodos necesarios para cargar y utilizar el modelo. La capacidad de este paquete para trabajar con embeddings de texto fue crucial para el proceso de similitud entre ofertas y demandas.

MySQL Connector: MySQL Connector fue utilizado para simplificar la conexión con la base de datos MySQL. Esta librería facilitó la ejecución de consultas y la gestión de la interacción entre la aplicación y la base de datos, garantizando una manipulación eficiente de los datos almacenados.

Logging[34]**:** La inclusión del paquete de Logging fue fundamental para documentar todas las actividades ocurridas en la aplicación en un archivo externo. Este registro resultó invaluable para el control de errores, ya que cualquier fallo o interrupción del servidor quedó documentado de manera detallada en archivos, proporcionando una herramienta valiosa para el análisis y solución de problemas.

4.5.4 - Modelo utilizado

Luego de una exhaustiva reunión con el codirector del proyecto, Gustavo Meschino, donde se evaluó la tecnología más acorde para solucionar el problema del sistema de matcheo, se presentaron las siguientes alternativas para el desarrollo de esta solución:

- **TF-IDF**[35]: Es una técnica clásica de procesamiento del lenguaje natural (NLP[36]) basada en la frecuencia de palabras, sin el uso de redes neuronales o aprendizaje profundo. No utiliza representaciones contextuales de las palabras.
    - **Ventajas**: Es una técnica probada y rápida, especialmente adecuada para tareas donde no se requiere una comprensión profunda del contexto.
    - **Limitaciones**: No captura adecuadamente el contexto de las palabras, ya que se basa únicamente en la frecuencia de palabras individuales sin tener en cuenta las relaciones semánticas. Esto hace que no sea adecuado para tareas que requieran un alto nivel de comprensión del lenguaje.
- **Modelos basados en reglas o heurísticas**[37]: Estos son métodos que no utilizan ninguna técnica de aprendizaje profundo ni redes neuronales. Se basan en reglas definidas manualmente para determinar la similitud entre textos, lo cual es una aproximación completamente diferente a tecnologías actuales, como, por ejemplo, los Transformers (que se explican en los siguientes párrafos).
    - **Ventajas**: Son métodos interpretables y fáciles de ajustar, ya que las reglas se pueden definir y modificar manualmente según las necesidades específicas de la aplicación.

    - **Limitaciones**: Pueden ser demasiado rígidos y carecen de la flexibilidad necesaria para manejar variaciones complejas en el lenguaje natural. Además, son difíciles de escalar y mantener a medida que crece la cantidad de datos y las variaciones en los textos.
- **Modelos basados en Sentence Transformers:** Los Transformers[38] son redes neuronales que pueden procesar secuencias ordenadas de información, aprenden contexto y, por lo tanto, significado mediante el seguimiento de relaciones en datos secuenciales como las palabras de una oración (representadas en forma de vectores) (Fúquene Ardila, 2024). Es decir, tratan de resolver el problema que ha tenido la inteligencia artificial por años, que es que puedan entender el contexto de una oración. Estas técnicas suelen ser más costosas en términos computacionales en comparación a las otras, pero dado que no se necesitaban respuestas en tiempo real, se desconsidera esta limitación. Luego, se plantearon dos alternativas:
    - **Utilizar un modelo pre entrenado:**
        - **Ventajas:** Usar un modelo pre entrenado ahorra tiempo y recursos, ya que se basa en modelos que ya han aprendido de grandes conjuntos de datos y solo necesitan ajustes mínimos para tareas específicas. Esto nos ayudaría a reducir el tiempo de desarrollo del proyecto.
        - **Limitaciones:** Los modelos pre entrenados pueden tener limitaciones si los datos con los que se entrenaron originalmente no son suficientemente relevantes para el dominio específico de la aplicación, que en cierto punto, podría llegar a ser un problema para el desarrollo.
    - **Construir y entrenar un modelo basado en sentence transformers:**
        - **Ventajas:** Se tendrá flexibilidad y el control total al entrenar nuestro propio modelo, lo que permite una personalización precisa para las necesidades específicas del proyecto. Este enfoque es ideal si se cuenta con datos específicos y se desea optimizar el rendimiento del modelo para un dominio concreto.
        - **Limitaciones:** Este enfoque es el más costoso en términos de tiempo y recursos computacionales. Además, requiere un gran conjunto de datos etiquetados y una infraestructura adecuada para entrenar el modelo desde cero, lo cual termina siendo un gran obstáculo para el desarrollo.

Luego de considerar estas opciones, se optó por utilizar un modelo pre entrenado basado en Sentence Transformers, debido a que se consideró que debido a la poca

cantidad de datos de prueba que se tendrán. Se consideró conveniente la elección de un modelo pre entrenado con una gran cantidad de datos, para luego hacerle un fine-tunning (transfer learning) modificando algunos parámetros y utilizando los datos adicionales recopilados.

Como se mencionó, estos Transformers tienen la capacidad de captar la semántica de un texto, representando palabras mediante vectores que contienen características específicas de cada palabra en el texto. Dado que los sistemas computacionales procesan números y no texto, la vectorización de palabras facilita a los sistemas la comprensión de palabras, transformándolas en valores. Este proceso se conoce como "text embedding[39]".

Los transformers se emplean principalmente en tareas relacionadas con el Procesamiento del Lenguaje Natural (NLP, por sus siglas en inglés). En este contexto, el desafío consistía en lograr la similitud entre la sentencia (el texto) de una oferta y el correspondiente a una demanda. La estrategia adoptada fue utilizar textos descriptivos de ambas y comparar sus similitudes para calculando un coeficiente de similitud mediante la distancia coseno. Esta aplicación específica de los modelos se denomina "sentence similarity" (similitud entre textos).

Existen modelos pre entrenados que extraen características de las palabras, describiéndolas con vectores de N dimensiones. Cuantas más dimensiones haya, mayor precisión se logra en la descripción de una palabra, aunque también aumenta la complejidad del análisis. Como ejemplo contextual (sin intentar determinar los detalles computacionales de estos modelos), la palabra "Gato" podría representarse con un vector de cinco posiciones, cada una correspondiente a un feature potencial, como la humanidad, la condición de animal, ser vivo, máquina y tamaño. Durante el entrenamiento de estos modelos, los pesos de las palabras se ajustan en estas posiciones según una gran cantidad de sentencias de entrenamiento. Esto se aprecia en la *Figura 8*, donde se muestra que el modelo le asigna estos coeficientes a cada posición dependiendo del dataset utilizado.

El resultado técnico de este proceso coloca a las palabras en un espacio vectorial donde se pueden realizar operaciones con ellas. En un dataset específico sobre animales, las palabras "Perro" y "Elefante" no estarán cercanas en el espacio vectorial, pero en un dataset genérico, al ser ambos animales, estarán más próximas. Este enfoque permite a los sistemas computacionales comprender la semántica de los textos, dando un paso más en dirección a cómo lo interpretan los humanos.

Otros modelos, como últimamente son los Transformers, son capaces de analizar secuencias de vectores. Si las palabras han sido convertidas en vectores, entonces los Transformers pueden analizar secuencias de palabras. Los modelos de este tipo incluyen una etapa de codificación que los hacen capaces de convertir una secuencia de palabras en un único vector de características. Es decir, luego de su procesamiento, es posible representar un texto o secuencia completa como un único vector de características. Estas características no son interpretables, pero permiten comparar entonces dos textos y cuantificar su similaridad.

Se utiliza la similitud del coseno para evaluar cuánta similitud existe entre dos vectores, como se ilustra en la “ecuación 1”. Este enfoque proporciona una métrica precisa para medir la relación semántica entre dos textos.

$$\frac{\overline{u}.\overline{v}}{|u|.|v|} = Similarity$$

Ecuación 1 - Cálculo de la similitud del coseno entre dos vectores

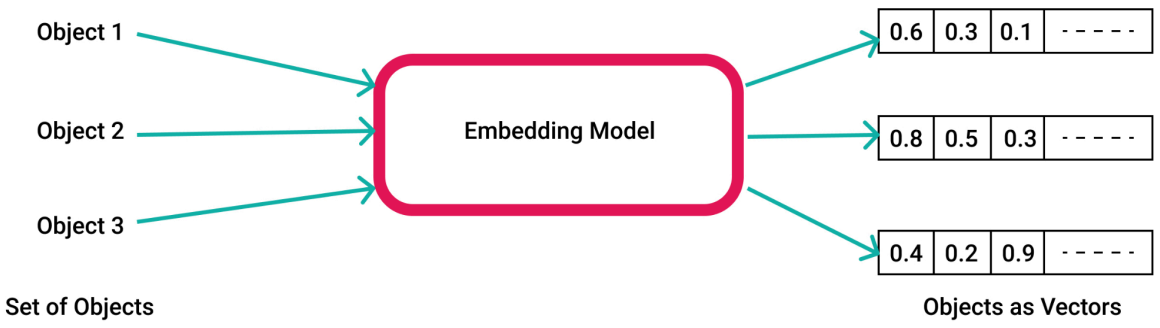

Figura 8 - Representación del ‘text-embedding’, por OnFinance AI, 2024

4.5.5 - Implementación

Para implementar este sistema de matcheo, se empleó un modelo pre entrenado de la biblioteca "Hugging Face[40]" denominado "all-mpnet-base-v2[41]" que implementa lo explicado en la sección anterior. Este modelo ya viene entrenado con vastos volúmenes de texto en varios idiomas, lo que implica que cuenta con embeddings predefinidos.

El modelo, al ya estar pre entrenado, tiene un tamaño de embedding predefinido, que es de dimensión 768. Volviendo al ejemplo analizado en la *Figura 8,* esto significa que los vectores que representan palabras tendrán 768 dimensiones. Por lo que cada palabra ingresada, se codifica como un vector de 768 dimensiones.

Para adaptar el modelo a nuestro contexto específico, fue necesario realizar un ajuste fino (fine-tuning). Este proceso fue adecuado y permitió mejorar su funcionamiento. Para lograrlo, se debió entrenar el modelo con datos provistos por los demandantes del proyecto, en este caso ofertas y demandas. Para ello, se agruparon las ofertas con demandas que se sabía de antemano que tenían algún tipo de relación, y se las calificó con un coeficiente de similitud. Luego, estas fueron ingresadas en el modelo ya mencionado para poder entrenarlo y conseguir que el modelo pueda adaptarse a nuestro contexto. Esto quiere decir, que el modelo mejora la forma de calcular los embedding, y nos proporciona unos vectores más precisos para que luego sean utilizados para calcular la similitud entre dos textos, y obtener así mejores resultados.

El fine-tunning debe ser coherente con el modelo pre entrenado, así que se mantuvo la forma de codificar la palabra, dejando un tamaño de embedding de 768 dimensiones. Luego, el resto de los hiperparametros se fijó en función de la documentación consultada. En la sección entrenamiento, se detalla el conjunto de datos utilizado.

Para interactuar con el modelo, se creó una API utilizando el paquete FastAPI. Esta API cuenta con dos endpoints principales: "/predict-offer" y "/predict-demand", mediante los cuales el backend consulta los matches para una oferta o demanda recién agregada. Cada vez que se llama a estos endpoints, el modelo recibe los inputs correspondientes, que son los textos descriptivos de una oferta o demanda. Luego, consulta la base de datos para obtener todas las otras ofertas o demandas y realizar comparaciones con la recién agregada.

La información obtenida se utiliza para calcular la similitud entre las ofertas y demandas. Se definió un umbral inicial de 0.7, el cual es configurable, basado en pruebas empíricas y recomendaciones estándar en el campo de modelos de similitud de texto. Si la similitud calculada supera este umbral, se considera que hay un match efectivo y el modelo devuelve al backend el ID del match realizado.

En los casos donde el coeficiente de similitud se encuentra entre 0.5 y 0.7, se implementó una segunda verificación basada en la cantidad de tags coincidentes entre las ofertas y demandas. Si hay coincidencia en al menos dos tags, se considera un match y se

trata la oferta-demanda como válida. Por último, si el coeficiente es inferior a 0.5, se descarta completamente el match, y se notifica al backend que realizó la petición.

Este enfoque híbrido permite un balance adecuado entre precisión y flexibilidad, asegurando que el sistema sea lo suficientemente sensible para captar matches relevantes, pero a la vez selectivo para evitar asociaciones incorrectas. Los valores de los umbrales fueron determinados de manera cuidadosa, considerando las características específicas del dataset y la necesidad de optimizar el rendimiento del sistema Puente.

El flujo del match se puede apreciar en la sección Anexo IX, donde se cómo es el proceso de matcheo, y a que atributos se acceden en la base de datos.

4.5.6 - Entrenamiento

Se detalla a continuación cómo se llevó a cabo el entrenamiento del modelo durante la etapa de fine-tuning. El fine-tunning se llevo a cabo siguiendo la siguiente documentación[42].

El principal desafío enfrentado fue la falta de una cantidad suficiente de ofertas y demandas en el conjunto de datos. Por este motivo, optamos por utilizar un modelo preentrenado y realizar un fine-tuning para adaptarlo al conjunto de datos disponible.

Dado que la plataforma se había desarrollado antes de implementar el módulo de inteligencia artificial, se permitió la carga de ofertas y demandas por parte del público. Esto tenía como objetivo incrementar el número de datos disponibles para el entrenamiento del modelo. De estos datos, lo más importante eran las descripciones de las ofertas y demandas.

Siguiendo lo mencionado anteriormente, se establecieron los hiperparámetros del modelo basándonos en la documentación consultada, donde se recomendaban ciertos valores adecuados para conjuntos de datos pequeños.

Después de un mes de esperar a que los usuarios cargaran sus ofertas y demandas, se obtuvieron un total de 32 ofertas y 2 demandas. Esta situación generó una incertidumbre significativa, ya que era evidente que el conjunto de datos seguía siendo insuficiente para un entrenamiento efectivo. Ante esta problemática, se decidió utilizar el generador de texto

ChatGPT para crear al menos 20 demandas adicionales a partir de las existentes, con el fin de ampliar el conjunto de datos.

Dado el conjunto de datos ampliado, se procedió a adaptarlo a las especificaciones requeridas por el modelo. En este caso, se generaron 704 ejemplos de entrenamiento, obtenidos al comparar cada oferta con cada demanda. El modelo requería que para cada par oferta-demanda se proporcionara un coeficiente normalizado (entre 0 y 1) que indicara el grado de similitud entre ambos textos.

Para organizar la información de manera estructurada, creamos tres archivos de texto con los siguientes formatos:

- **Ofertas.txt**: Contiene las ofertas siguiendo el formato:

  ```
  {id_oferta, texto}
  ```

- **Demandas.txt**: Contiene las demandas siguiendo el formato:

  ```
  {id_demanda, texto}
  ```

- **matches.txt**: Contiene los pares oferta-demanda junto con su coeficiente de similitud, en el formato:

  ```
  {id_oferta, id_demanda, coeficiente}
  ```

En base a esto, se optó por dividir el dataset en 80% datos de entrenamiento y el 20% restante a datos de prueba para la evaluación del modelo, quedando 563 datos de entrenamiento, y 141 para datos de test.

Finalmente, con el conjunto de datos preparado, procedimos al entrenamiento del modelo utilizando estos ejemplos.

Como resultado, se utilizaron los 141 datos para el testeo a mano del sistema de matcheo. La evaluación de este sistema, fue corroborar que los matcheos propuestos correspondan correctamente a lo que se esperaba, lo cual ocurrió exitosamente. Se especifica en la sección trabajo futuro el encontrar medidas más apropiadas para la evaluación del sistema, debido a que excede el *scoope* de los objetivos propuestos en este trabajo.

4.6 - Base de datos

4.6.1 - Tecnologías

Para la elección de la base de datos, se inició consultando con el administrador del servidor de la Universidad para conocer qué motores de bases de datos estaban disponibles para su uso. Se descartó la opción de Oracle[43], ya que su licencia implica costos y no estaba disponible para un uso completo en este contexto.

En los servidores de la Universidad, se contaba con la opción de utilizar MariaDB[44], que es compatible con MySQL pero ofrece una versión completamente gratuita y de código abierto. Dado que ambos sistemas comparten una alta compatibilidad, se optó por la comodidad y familiaridad, desarrollando la base de datos con MySQL 8.0.

4.6.2 - Diagrama Entidad - Relación

Como buena práctica a la hora de diseñar un sistema de base de datos, luego de analizar el problema, se procedió a hacer el diagrama de entidad relación (DER[45]) . En este caso, el desarrollador ya contaba con experiencia desarrollando sistemas de bases de datos, además de tener los conocimientos recientes de la materia, donde se lleva a cabo un proceso completo del diseño de una base de datos. El DER puede encontrarse en la sección Anexo III: Diagrama Entidad Relación

En general, al aplicar la metodología del Diagrama Entidad-Relación (DER) y con la experiencia del desarrollador en la asignación de atributos a cada entidad, se suele lograr un buen diseño de base de datos, sin necesidad de analizar explícitamente las formas normales que deben cumplir las tablas. Aunque no se realice un análisis detallado de la normalización, este aspecto se considera de manera implícita durante el proceso.

Una vez completado el análisis inicial, la base de datos resultante sigue ciertas reglas de diseño que ayudan a su estructura, aunque es posible que no sea completamente óptima en términos de tiempos de acceso. Por eso, una vez terminada esta etapa, se evalúan los atributos de las tablas que son más frecuentemente consultados. Esta evaluación permite determinar si es conveniente desnormalizar la base de datos, añadiendo atributos que puedan ser redundantes pero que, en la práctica, mejoren el rendimiento del sistema.

En este caso específico, no se analizó a fondo la normalización de cada tabla, pero se priorizó inicialmente utilizar solo los atributos necesarios para mantener la base de datos

ligera y fácil de entender. Posteriormente, según las necesidades de rendimiento, se añadieron atributos adicionales cuando se consideró oportuno hacerlo.

En la sección "Anexos: Diagrama Entidad-Relación" se proporciona el archivo con el diagrama completo.

4.6.3 - Métricas de la base de datos

Se hizo un breve análisis de métricas de la base de datos, que nos permite visualizar sí la misma tiene un tamaño significativo o no. Además, también nos permite ver que si tenemos pocas filas en una tabla pero su peso es muy grande, quizás se deba definir la estructura de la base de datos. La tabla generada es la siguiente:

Nombre de la tabla	Cantidad de filas	Cantidad de columnas	Peso de la tabla [KB]
demanda	2	10	32.0
demanda_perfiles	4	3	48.0
demanda_segmentos	3	3	48.0
demanda_tags	8	3	32.0
dominios_permitidos	2	2	16.0
linea_financiamiento_beneficiarios	11	2	16.0
linea_financiamiento_secciones	38	4	80.0
linea_financiamiento_tipo	3	2	16.0
lineas_financiamiento	8	10	48.0
matches	2	4	48.0
oferta_faqs	91	4	96.0
ofertas	67	10	144.0
ofertas_contactos	2	11	32.0
ofertas_estados	207	6	48.0

ofertas_imagenes	100	4	32.0
ofertas_personas	104	7	32.0
ofertas_segmentos	335	3	48.0
ofertas_tags	406	3	64.0
ofertas_tipos	117	3	48.0
ofertas_visualizaciones	869	3	80.0
perfiles	9	2	16.0
roles	5	3	16.0
segmentos	21	2	32.0
tipos_oferta	4	2	16.0
ue_nact	4	2	16.0
unidad_ejecutora	240	7	96.0
usuario_roles	82	3	48.0
usuario_ue	74	4	48.0
usuarios	72	8	64.0
TOTAL	**2890**	**130**	**1.328**

Tabla 1 - Métricas de la Base de Datos

Como se puede observar en la *Tabla 1* , el tamaño de la tabla de usuarios es de 72 registros, la tabla de ofertas cuenta con 67 ofertas (tanto disponibles como no disponibles), y la tabla de demandas tiene solo dos demandas. Hasta el momento, la base de datos tiene un número reducido de entradas, por lo que, en principio, no debería presentar problemas de rendimiento. En la sección trabajo futuro, queda aclarado que cuando la base de datos escale, se deberá nuevamente hacer un análisis de las estadísticas y su rendimiento para determinar las acciones a seguir.

4.7 - SMTP - Envío de mails

Uno de los requisitos esenciales delineados por el referente funcional consiste en la implementación de un sistema de notificación por correo electrónico en respuesta a ciertos eventos específicos. Estos eventos engloban situaciones tales como:

- Alerta de oferta pendiente de autorización a los administradores correspondientes.
- Comunicación de rechazo o autorización de una oferta a un usuario oferente.
- Notificación de matcheo entre oferta y demanda dirigida a usuarios oferentes o demandantes.
- Aviso de contacto entre un demandante y un oferente.

En respuesta a este requerimiento, se consideró la infraestructura tecnológica de la Universidad, destacando la presencia de un servidor SMTP. Este servidor facilita la creación de dominios personalizados, como el dominio "@unmdp.edu.ar". Se obtuvo un usuario específico, puente@mdp.edu.ar, con la finalidad de gestionar los envíos de correos electrónicos. El backend, para cumplir con esta funcionalidad, se vale del paquete de terceros llamado "nodemailer". Este paquete se encarga de establecer la comunicación con el servidor SMTP utilizando las credenciales proporcionadas, permitiendo así la efectiva implementación del sistema de notificación por correo electrónico.

4.8 - Seguridad

La seguridad constituye un elemento fundamental en el desarrollo de una plataforma. Dado que se maneja información sensible de los usuarios, es crucial otorgar prioridad a un sólido manejo de la seguridad. Se implementaron medidas exhaustivas en todos los entornos, tanto en el backend como en el frontend, con el objetivo de maximizar la seguridad de la plataforma. A continuación, se destacarán los aspectos más significativos relacionados con la seguridad de la plataforma.

4.8.1 - JSON WEB TOKEN + BCRYPT

Para asegurar la comunicación segura entre el cliente, el servidor y la base de datos, se implementaron técnicas basadas en tokens para verificar la validez y la autoridad de una sesión. Esto significa que un usuario solo puede acceder a una parte privada de la aplicación si posee un token válido y, al decodificarlo, tiene los roles necesarios para el acceso.

La implementación se basó en JSON Web Token (JWT[46]), un paquete de terceros diseñado para generar tokens a partir de información proporcionada. Cuando un usuario se autentica con sus credenciales, el servidor responde con un token en los encabezados de la respuesta. El usuario almacena este token y lo utiliza en todas las solicitudes en las que sea necesario. Cada vez que se solicita un recurso, el servidor procesa y decodifica el token para verificar su validez y asegurarse de que cumple con los roles necesarios para acceder al recurso. El token se cifra mediante una clave secreta de firma, almacenada en variables de entorno y conocida solo en el servidor. Esta clave firma los tokens al emitirse y los descifra al procesarlos en una solicitud. El token contiene información como el ID del usuario, nombre, apellido y roles.

Para el almacenamiento seguro de contraseñas, se utilizó el módulo de terceros "bcryptjs[47]". Este paquete utiliza un algoritmo de encriptación desconocido y genera un hash único para cada contraseña de usuario proporcionada. Además, se proporciona una "salt" o semilla, un valor aleatorio único que se combina con la contraseña antes de realizar el hash. La sal agrega entropía al proceso de hashing, lo que garantiza que incluso si dos usuarios tienen la misma contraseña, sus hashes serán diferentes debido a las sales únicas. Luego, la contraseña almacenada con hash se recupera y se compara con la nueva contraseña ingresada por el usuario al iniciar sesión, evaluando los hashes en lugar de desencriptar la contraseña original.

Finalmente, el backend implementó un endpoint con la ruta "/validate" que invoca un método, también un middleware, para validar la autenticidad del token. Esta solicitud se realiza desde el frontend en varias instancias, como al cambiar de ruta, como se detallará más adelante.

4.8.2 - Variables de entorno

En el servidor y en el frontend, se emplean variables de entorno con el propósito de maximizar la protección de información sensible, como contraseñas de bases de datos y claves secretas de desencriptación. Este enfoque resulta beneficioso al evitar la inclusión directa de estas claves en el código fuente de la aplicación, reduciendo así el riesgo de exposición, ya sea de manera accidental o intencionada. De esta manera, se fortalece la seguridad al separar la información confidencial de la lógica del programa y al proporcionar una capa adicional de resguardo.

4.8.2 - AuthGuards

En el cliente, se implementó la protección de rutas, complementando la seguridad proporcionada por el servidor. Esto asegura que usuarios malintencionados no puedan acceder a rutas sin autorización mediante la manipulación de URLs. Para este propósito, se utilizaron los AuthGuards[48] de Angular, interceptores de rutas que garantizan el acceso solo a usuarios autenticados y autorizados.

Cuando un usuario intenta acceder a una ruta protegida, el AuthGuard verifica su autenticación y permisos. Esta medida se complementa con una petición al backend a la ruta "/validate", como se explicó anteriormente. En cada cambio de ruta, se envía una solicitud a este endpoint desde un AuthGuard. Este último gestiona la respuesta y, en caso de una autorización fallida, cierra la sesión del usuario y lo redirige a la página de inicio de sesión.

4.8.3 - Roles

La plataforma implementa un sistema de roles para gestionar los niveles de acceso a diversos recursos. La asignación de roles se realiza mediante un administrador general, creado directamente en la base de datos. Este administrador tiene la capacidad de gestionar los roles de otros usuarios a través de la interfaz de la aplicación, asegurando que solo los usuarios autorizados puedan llevar a cabo acciones protegidas. Los roles disponibles en la plataforma son los siguientes:

- Administrador General:
    - Encargado de gestionar los privilegios de los usuarios, como se mencionó anteriormente.

- Administrador Rectorado:
    - Usuario gestionado por el rectorado de la Universidad.
    - Autoriza ofertas en nivel 2.
    - Visualiza las ofertas pendientes en nivel 1.
    - Accede a información sobre los matches de distintas ofertas y sus contactos.

- Administrador Unidad Oferente:
    - Usuario asignado a los directores de grupos.

    - Autoriza ofertas a nivel 1.
    - Carga líneas de financiamiento.

- Oferente:
    - Usuario con correo con dominio facultativo: @unmdp.edu.ar.
    - Gestiona sus ofertas.
    - Visualiza las demandas disponibles.

- Demandante:
    - Usuario fuera del sistema facultativo, destinado al sector empresarial.
    - Gestiona sus demandas.
    - Visualiza las ofertas disponibles.

Este sistema de roles garantiza que cada usuario tenga acceso y capacidad de gestión acorde con sus responsabilidades y funciones en la plataforma.

4.8.4 - API Sistema de Matcheo

En el servicio del sistema de matcheo, se ha implementado una API dedicada para gestionar las solicitudes relacionadas con dicho sistema. Dado que esta construcción no es extensa y solo es consultada internamente desde el backend, no se requirieron medidas de seguridad extensas. No obstante, se ha optado por restringir el acceso a las solicitudes únicamente desde el localhost, es decir, desde el servidor de backend (alojado en la misma dirección que la API del sistema de matcheo). Además, el puerto utilizado por este servicio no está expuesto y no es accesible desde fuera, proporcionando así una capa adicional de seguridad.

4.8.5 - Ataques

Para finalizar la sección de “Seguridad”, se elaboró un documento que se entregará al referente funcional, con los posibles ataques y las soluciones provistas en Puente para ello. Puede verlo en la sección: Apéndice: Documento contra ataques.

PLATAFORMA PUENTE
DOCUMENTACION SOBRE POSIBLES ATAQUES

ATAQUE	EXPLICACIÓN	SOLUCIÓN	DOCUMENTACIÓN
Inyección SQL	Las inyecciones de SQL (o SQLI) se producen cuand	El backend de la plataforma puente esta desarrolla	ps://github.com/mysqljs/mysql#escaping-query-valu
Inyección HTML Inyección JAVASCRIPT	Las inyecciones HTML o de SCRIPTS son un tipo de	Puente esta desarrollado mediante la tecnología a	https://angular.io/guide/security
Ataques de autenticación	Los atacantes podrian vulnerar la base de datos y ob	El sistema implementa una encriptación de las con	No aplica
Exposición de archivos sensibles:	Los archivos de configuración del proyecto puede qu	Utilización de variables de entorno del sistema, do	No aplica
Intercepción de información	El uso de protocolo sin encriptación puede lograr que	Uso de protocolo HTTPS para encriptación de los	No aplica
Vulnerabilidad de claves de tokens	Los tokens que no viajen protegidos por header pued	En Puente, todos los token de autenticacion viajan	No aplica

PLATAFORMA de VINCULACIÓN TECNOLÓGICA
UNIVERSIDAD NACIONAL DE MAR DEL PLATA
PUENTE

Figura 9. Documentación contra ataques

4.9 - Testing

Se ha empleado Postman[49] para llevar a cabo pruebas de los endpoints en el backend. La elección de esta herramienta se basa en su popularidad y en sus características avanzadas para la evaluación exhaustiva de un backend. Postman no solo permite probar los servicios, sino que también proporciona herramientas adicionales para automatizar llamadas a los servicios y ejecutar acciones basadas en las respuestas obtenidas. La preferencia por Postman en lugar de otras herramientas, como Thunder, se debió a la experiencia previa del desarrollador con Postman.

Para las pruebas en el frontend, se recurrió al uso de las consolas de desarrollo y se llevaron a cabo pruebas de funcionalidad cada vez que se implementaba un nuevo módulo. Además, se adoptó un enfoque proactivo al considerar combinaciones inusuales, como caracteres especiales u otros inputs no deseados, con el objetivo de prevenir posibles errores. Este enfoque asegura una evaluación exhaustiva de la robustez y funcionalidad del frontend, contribuyendo a la detección y corrección temprana de posibles problemas durante el desarrollo.

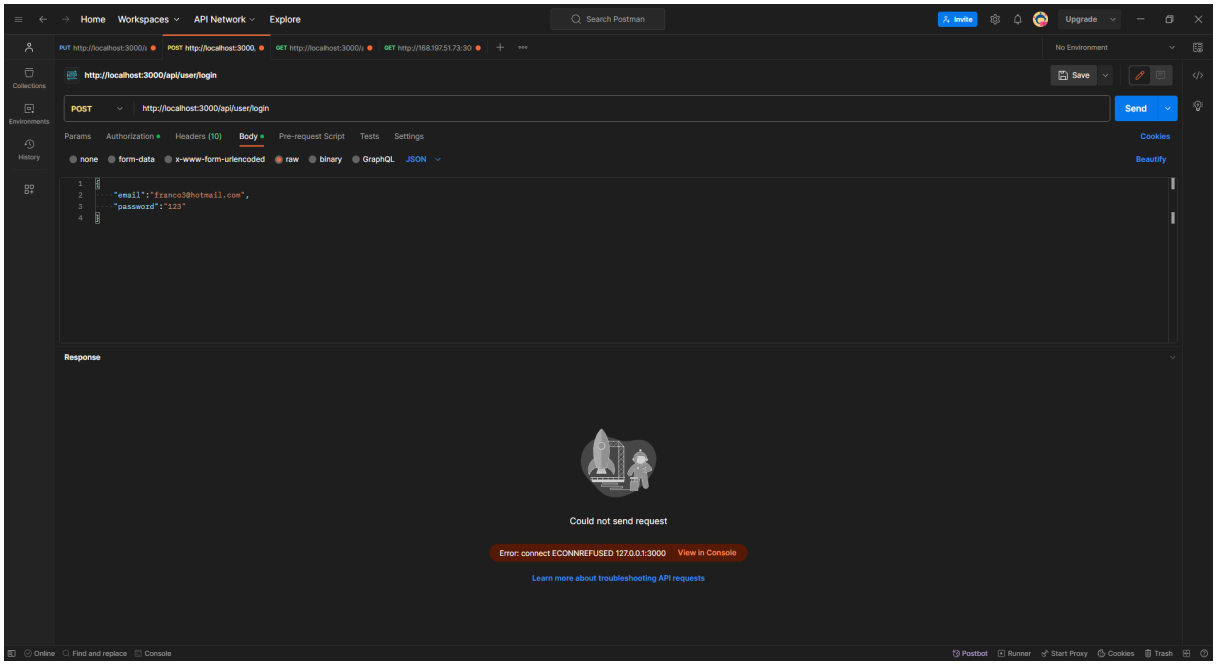

Figura 10. Interfaz principal de 'Postman'

4.10 - Documentación

La documentación desempeña un papel fundamental en la construcción de aplicaciones escalables, ya que proporciona información esencial sobre la implementación, facilitando la reutilización y modificación del código. En este contexto, los métodos en el código fuente del frontend están documentados directamente en el código, mediante comentarios descriptivos.

En cuanto a las API, se ha adoptado la práctica común de construir una "API-DOCS" para comprender a fondo la funcionalidad de cada endpoint. Para este propósito, en Puente, se ha empleado el paquete de terceros "SwaggerJS" para la documentación completa de la API. Swagger permite documentar exhaustivamente cada endpoint del backend mediante comentarios, y proporciona una interfaz gráfica que reside en el mismo servidor.

Para acceder a la API, se puede dirigir a la sección correspondiente. La API se crea junto con el servidor, y también se puede acceder a ella mediante la ruta "dominio/api-docs". Sin embargo, para garantizar la seguridad y evitar exponer la API al público en producción, se ha decidido mantenerla oculta. Esto significa que, al ejecutar el código en un entorno de desarrollo, al dirigirse a "dominio/api-docs", se encuentra la interfaz que contiene toda la

documentación detallada de los endpoints del backend. Por motivos de seguridad, en el entorno de producción, se ocultó la posibilidad de ingresar a la documentación de Puente.

La utilización de Swagger no solo proporciona una documentación clara y completa, sino que también facilita la comprensión de la API a través de una interfaz gráfica intuitiva, mejorando la eficiencia y la experiencia del desarrollo.

En la sección 'Apéndice: *Documentación API REST*' se puede encontrar la documentación completa de manera offline.

4.11 - Backups

Con el objetivo de prevenir posibles pérdidas de información, se ha implementado un script automatizado que se ejecuta periódicamente, guardando copias de seguridad de la base de datos en el servidor. Este enfoque es fundamental para mitigar cualquier error o pérdida involuntaria de información.

El script programado está diseñado para ejecutarse a intervalos regulares (cada 7 días) y realiza la tarea crucial de respaldar la base de datos. Esta práctica asegura la disponibilidad de versiones anteriores de los datos, lo que resulta esencial en situaciones donde se pueda cometer un error o se elimine información de manera inadvertida.

La implementación de este proceso automatizado no solo contribuye a la prevención de pérdidas de datos, sino que también proporciona una capa adicional de seguridad y confiabilidad al garantizar la disponibilidad de copias de seguridad actualizadas en el servidor. Este respaldo periódico se convierte en un salvaguarda vital para la integridad y la recuperación de datos en caso de eventos imprevistos o errores en la base de datos.

El script generado fue el siguiente: 'Apéndice: Script para backup', y se generó mediante el uso de un archivo de extensión '.sh' para la ejecución del mismo en terminal. Luego, el comando utiliza 'mysqldump[50]' que permite ejecutar querys sql como en este caso es el backup.

En este caso, el deploy fue en un sistema que utilizaba linux, por lo que se utilizó el archivo 'crontab[51]' que poseen los sistemas basados en linux para poder programar una ejecución de este comando cada 7 días.

4.12 - Problemáticas

En el transcurso del desarrollo del proyecto se generaron múltiples problemáticas. A fines prácticos, se clasificaron en 3 grupos:

- **Técnicas:**

  Durante la etapa de desarrollo, surgieron diversas problemáticas, tanto centradas en el código como problemas con el hosting donde se alojaba temporalmente la plataforma. Para ello los servicios proveedores de hosting proporcionan un soporte en línea donde se pueden ingresar consultas para que el soporte responda. En varias ocasiones, los problemas se podían solucionar sencillamente, pero en otras, no se pudo solucionar y se debió cambiar el hosting. Algunas problemáticas fueron:

  Problemáticas con hostings: Durante la fase de desarrollo, surgieron diversas problemáticas que abarcaban tanto aspectos relacionados con el código como inconvenientes con el servicio de hosting temporal utilizado para la plataforma. Para abordar estas cuestiones, se recurrió al soporte en línea proporcionado por los servicios de hosting, donde se presentaban consultas para recibir asistencia. En varios casos, las problemáticas pudieron resolverse de manera sencilla, mientras que en otros casos, la solución implicó cambiar de proveedor de hosting.

  Error de Conexión con SMTP: Se enfrentó a una situación en la que no se pudo resolver un error de conexión entre el servidor de hosting y el servidor SMTP de la Universidad. Esto condujo a la decisión de cambiar de proveedor de hosting.

  Despliegue del Backend: Se experimentaron dificultades inicialmente al intentar desplegar el backend debido a la falta de una interfaz proporcionada por el hosting. Posteriormente, se aclaró que se requería la contratación de un plan específico para aplicaciones con Node.js.

  Problemas con el Archivo .htaccess: Al desplegar una aplicación SPA (Single Page Application) en un servidor con Apache, surgió la necesidad de configurar un archivo llamado ".htaccess" para definir reglas de direccionamiento de la página. Dado que no se trataba de una aplicación con

SSR (Server Side Rendering), cada cambio de ruta no generaba una petición GET al servidor, sino que se mostraban u ocultaban componentes. Sin embargo, si el archivo ".htaccess" no se configuraba con reglas que indicaran que no debía realizarse una petición GET en cada cambio de ruta, la aplicación no funcionaría correctamente, ya que no tendría nada que solicitar al backend y devolvería un error 404 Not Found.

Configuración de VirtualHost[52]**:** Inicialmente, la plataforma no se encontraba alojada en los servidores de la Universidad debido a restricciones burocráticas, que impedirían cambios rápidos en caso de fallos. En consecuencia, optamos por redirigir temporalmente el subdominio proporcionado por la Universidad hacia un servidor externo contratado. Esto significa que al ingresar "puente.mdp.edu.ar", se redirige al servidor contratado. Para lograr esto, fue necesario configurar los VirtualHosts en el servidor Apache contratado y definir reglas de reescritura. Aunque esto generó problemas y retrasos en la entrega, se resolvieron con éxito.

Mantener Procesos Vivos en SSH[53]**:** Dado que era necesario utilizar SSH[54] para iniciar los procesos de los servidores, se enfrentaba el problema de que estos procesos finalizaban cada vez que se cerraba la consola. Para abordar este inconveniente, se investigó una solución para mantener los procesos activos y reiniciarlos, si fuera necesario, al cerrar la terminal. En este contexto, se empleó la herramienta "pm2", que permite mantener activos los procesos y reiniciarlos en caso de errores, ofreciendo así una mayor estabilidad y continuidad en la ejecución de los servicios.

Obtener Certificado para HTTPS: En la fase inicial, la plataforma estaba alojada en un servidor externo que proporcionaba un certificado a nombre de un dominio denominado "netgenprueba.com.ar". Sin embargo, al redirigir el tráfico, el certificado no abarcaba el dominio "puente.mdp.edu.ar". Esto generaba desconfianza, ya que los usuarios veían la falta de certificación al acceder al sitio. Para resolver este problema, se recurrió a "Let's Encrypt[55]" para generar un certificado específico para "puente.mdp.edu.ar". Let's Encrypt es una plataforma que produce certificados personalizados, válidos por un tiempo limitado, y ofrece este servicio de manera gratuita.

Entrenamiento red neuronal: Ya que no se contaban con las suficientes ofertas y demandas para lograr un buen dataset, se optó por una mejor solución. La misma contaba en mostrarle ejemplos a un generador de texto como ChatGPT, y que en base a las ofertas y demandas existentes, generará otras similares para ampliar el volumen del dataset, y así poder tener un resultado más relevante en el entrenamiento.

Despliegue en el servidor de la Universidad: En la Universidad, la seguridad es un aspecto significativo, especialmente cuando se trata de alojar aplicaciones en los servidores institucionales. Esta precaución tiene sentido, ya que garantiza la integridad de los servidores y previene la aparición de vulnerabilidades. Para llevar a cabo el despliegue de la aplicación Puente, la Secretaría de Vinculación y Transferencia tuvo que solicitar formalmente al área de servidores de la Universidad la preparación adecuada de los mismos. Inicialmente, se realizó un pedido especial para la instalación de NodeJS y Python, necesarios para ejecutar las API de la aplicación. Esto se debió a que los servidores estándar no contaban con estas tecnologías instaladas, dado que la mayoría de las aplicaciones alojadas en la universidad se desarrollan en PHP. Además, por razones de seguridad, el acceso SSH (Secure Shell) no está permitido. Esto implica que las dependencias de las aplicaciones solo pueden ser instaladas por un administrador de la universidad, lo cual resultó en demoras significativas en el despliegue final de la aplicación. Esta serie de procedimientos y restricciones reflejan el compromiso de la Universidad con la seguridad y la integridad de sus sistemas, aunque a veces pueda generar dificultades en el proceso de desarrollo y despliegue de aplicaciones. Todo lo mencionado, sumado al receso de verano, generó una demora considerable en el despliegue de la aplicación. Para las fechas acordadas, Puente ya estaba completamente desarrollada y alojada en un servidor externo a la Universidad, esperando el permiso para migrarla. Sin embargo, la burocracia y la resistencia a adoptar nuevas tecnologías en los servidores de la universidad provocaron retrasos en la migración. Para que la aplicación funcione con el dominio “puente.mdp.edu.ar” mientras se solucionaba el problema del despliegue en la Universidad, se nos ofreció tener la aplicación desplegada en el servidor de “DonWeb” momentáneamente, y que los servidores de la Universidad apunten mediante configuración del DNS[56]

hacia nuestro servidor externo. Esto fue lo que finalmente se hizo y durante los meses de receso, Puente funcionó de esta forma. Una vez finalizado el receso, se les ofreció una solución para el despliegue final de la aplicación mediante la utilización de Docker, lo que finalmente pudo llevarse a cabo. Esto se detallará en la sección "Despliegue: Servidor de la universidad".

- **Diseño:**

  General: Basándonos en los requerimientos establecidos, nos enfrentamos a un desafío al buscar un diseño que fuera moderno pero no demasiado disruptivo, dado que se trata de una plataforma asociada a una institución académica. En la actualidad, la tendencia en diseño se inclina hacia colores vibrantes, imágenes llamativas y abundancia de iconos, lo cual contrasta con la necesidad de mantener un cierto nivel de formalidad acorde con el contexto universitario. Esta situación se complicó aún más por el hecho de que el desarrollador no es un experto en diseño de experiencia de usuario (UX/UI). Superar esta dificultad implicó encontrar un equilibrio entre la estética moderna y la formalidad necesaria para una plataforma universitaria. Fue necesario explorar opciones que fueran visualmente atractivas pero también respetuosas con el entorno académico, lo que demandó un proceso de iteración y ajustes constantes hasta alcanzar un diseño que satisficiera las expectativas y necesidades del proyecto.

  Experiencia de usuario: Como se mencionó anteriormente, los usuarios tienen una experiencia limitada con sistemas web, por lo que fue crucial garantizar una experiencia de usuario intuitiva para que comprendieran claramente cómo utilizar la plataforma. Este aspecto representó un desafío significativo, especialmente porque el desarrollador carecía de experiencia en el diseño de experiencia de usuario (UX). La falta de experiencia en UX del desarrollador resultó en varios ajustes en la interfaz a lo largo del proceso de desarrollo. Fue necesario realizar cambios iterativos para simplificar la navegación, mejorar la usabilidad y asegurar que las funciones principales fueran accesibles y comprensibles para los usuarios menos experimentados.

- **Recursos humanos:**

Unificación de criterios en diseño y carga de datos: Debido a la diversidad de actores involucrados en el sistema, fue necesario alcanzar consensos tanto en el diseño de la aplicación como en la carga de datos. En cuanto al diseño, se solicitó a los usuarios que contribuyeron con ideas para el logo de la plataforma, lo que permitió establecer los colores principales. Este proceso desencadenó numerosas discusiones internas sobre el diseño del logo y la apariencia general de la plataforma, con el objetivo de causar una impresión específica en los usuarios. Estas deliberaciones prolongaron los tiempos de desarrollo. Además, surgieron conflictos relacionados con la carga de datos, en su mayoría entre los demandantes. A través de reuniones entre los secretarios de vinculación de las distintas unidades académicas, se debatió cómo agrupar los distintos grupos relacionados y qué información sería necesaria mostrar. Estos desafíos ocasionaron varios ajustes durante el desarrollo, lo que a su vez provocó demoras adicionales en el proceso.

Capacitación: La capacitación tuvo algunas complicaciones debido a la inexperiencia de algunos usuarios tratando con sistemas como este. Esto generó algunas quejas sobre mal funcionamiento o sobre faltante de funcionalidades cuando en realidad eran los mismos usuarios que no habían aprendido bien cómo utilizar la plataforma. Por fortuna, con el tiempo estas problemáticas fueron desapareciendo. En la sección “Capacitaciones” se explica más en detalle.

4.13 - Deploys

Los proveedores de hosting contratados para el desarrollo fueron dos, debido a que se inició con el primero y se cambió al segundo. Luego, el deploy final fue en los servidores de la Universidad. Ambos servidores tuvieron sus complicaciones para hacer los deploys, así que se detallará la forma de deployar en cada hosting:

NutHosting[57]**:** El servicio ofrecido era mediante la interfaz de CPanel[58]. Como política del hosting, no proveían acceso FTP[59] (File Transfer Protocol) ni SSH (Secure shell). Esto quiere decir que solamente se podían subir archivos mediante su File System, y solamente se podía hacer el deploy del backend en Node JS mediante su interfaz. Esto

imposibilitaba el encendido de un servicio y se supo que a futuro iba a traer problemas, ya que el desarrollo de la inteligencia artificial era mediante Python y se necesitaba iniciar un servicio y sin acceso SSH no era posible.

DonWeb[60]**:** El servicio ofrecido era mediante la interfaz de Ferozzo. Se ofrecían máquinas virtuales y su acceso tanto FTP como SSH. Esto, en un principio complicó el deploy debido a que el sistema operativo utilizado por la MV provista por DonWeb era Linux "CentOS 7[61]", por lo que se debió investigar sobre cómo manipular el sistema operativo y poder instalar la versión de node, mysql, y python correspondientes para poder ejecutarlos. Una vez se pudo lograr esto, fue beneficioso debido a la gran independencia que se tenía respecto del soporte, ya que se pudo usar a conveniencia todas las libertades que ofrece linux.

Servidor de la Universidad: El último despliegue se realizó de forma tardía, como ya se habló en la sección "Problemáticas". Debido a las tecnologías utilizadas por Puente (Node.js para el backend y Python para el sistema de matching), se decidió, por parte de las autoridades del área de servidores de la Universidad, que Puente sería alojado en un servidor ubicado en un departamento anexo a la universidad, situado en las calles San Martín y San Luis. Para ello, Franco Kuhn, responsable del mismo, nos ofreció acceso SSH y la posibilidad de utilizar Docker para facilitar el despliegue de la plataforma. El servidor asignado estaba vacío, por lo que primero se tuvo que instalar Docker y preparar el servidor para alojar la aplicación. En la sección "Docker", se explicará con mayor detalle cómo se llevó a cabo el despliegue.

Configuraciones en común

A pesar de que los proveedores de hosting fueron distintos, existían configuraciones en común que se detallarán en este apartado, especialmente en el Servidor de la Universidad y en el hosting DonWeb. Esto se debe a que ambos utilizaban máquinas virtuales, lo que nos otorgaba más libertad para la configuración de nuestro entorno. Las características son las siguientes:

Configuración de certificados SSL: Para la configuración de los certificados, se utilizó el módulo "certbot" de la autoridad certificadora "Let's Encrypt", que es gratuita y ampliamente utilizada en el mercado. Para ello, primero se instaló el módulo certbot en el entorno de producción y se configuró el servidor de Apache para que apuntara a estos certificados. Esto se detallará con precisión en la sección "Certificados SSL".

Configuración de VirtualHost: Los VirtualHost sirven para configurar un servidor de Apache. Aquí se indican configuraciones como el "Nombre del servidor", el "Alias del Servidor" y se generan las configuraciones de certificados. Lo que se hizo aquí fue generar configuraciones a nombre de "puente.mdp.edu.ar" y marcarlas como activas en el servidor para que pueda manejar las solicitudes dirigidas a ese dominio. Además, dentro de las configuraciones del VirtualHost, se agregó que todas las solicitudes dirigidas al puerto 80 (es decir, protocolo HTTP) sean redirigidas al puerto 443 (protocolo HTTPS).

Configuración de Proxy[62]**:** Debido a que la aplicación cuenta con un backend en Node.js, ejecutado en el mismo servidor donde está funcionando Apache, se debió encontrar una forma para que las solicitudes que fueran a /api se redireccionen a la aplicación de Node.js y no busquen un archivo estático de frontend. Todo esto se hizo para evitar exponer el puerto donde Node.js estaba funcionando y así no generar otra vulnerabilidad. Para ello, se optó por la configuración de Proxys inversos. Estos se configuran en los VirtualHost de Apache, donde con tan solo indicar las siguientes líneas en el archivo de configuración del mismo, todas las solicitudes que comiencen con /api serán redireccionadas a la aplicación de Node.js, y responderá en lugar de hacerlo el frontend. Las líneas son las siguientes:

ProxyPass /api https://localhost:3000/api
ProxyPassReverse /api https://localhost:3000/api

4.14 - Docker

4.14.1 - Introducción

Como se mencionó anteriormente, se decidió utilizar Docker para facilitar el despliegue final de la aplicación y no depender particularmente de los servidores ofrecidos. La idea surgió a partir de que el desarrollador observó la gran cantidad de dependencias y configuraciones que debía realizar en el servidor, por lo que, para evitar problemas futuros, se prefirió el uso de Docker[63].

Docker es una herramienta que genera contenedores a partir de lo que se llama una imagen, que puede ser tanto un sistema operativo como Ubuntu[64], o una imagen que

contenga únicamente un lenguaje de programación listo para su ejecución. La finalidad es evitar los problemas conocidos en el ámbito como "En mi máquina funciona". Docker logra encapsular un entorno elegido por el desarrollador dentro de un contenedor, con las funcionalidades mínimas para hacerlo óptimo, para luego instalar las dependencias necesarias y poder ejecutarlo. Una vez generado el contenedor, este puede ser trasladado de diversas formas a otra máquina, donde podrá ser ejecutado y funcionar de manera similar a como lo hacía en la máquina donde fue configurado.

4.14.2 - Arquitectura

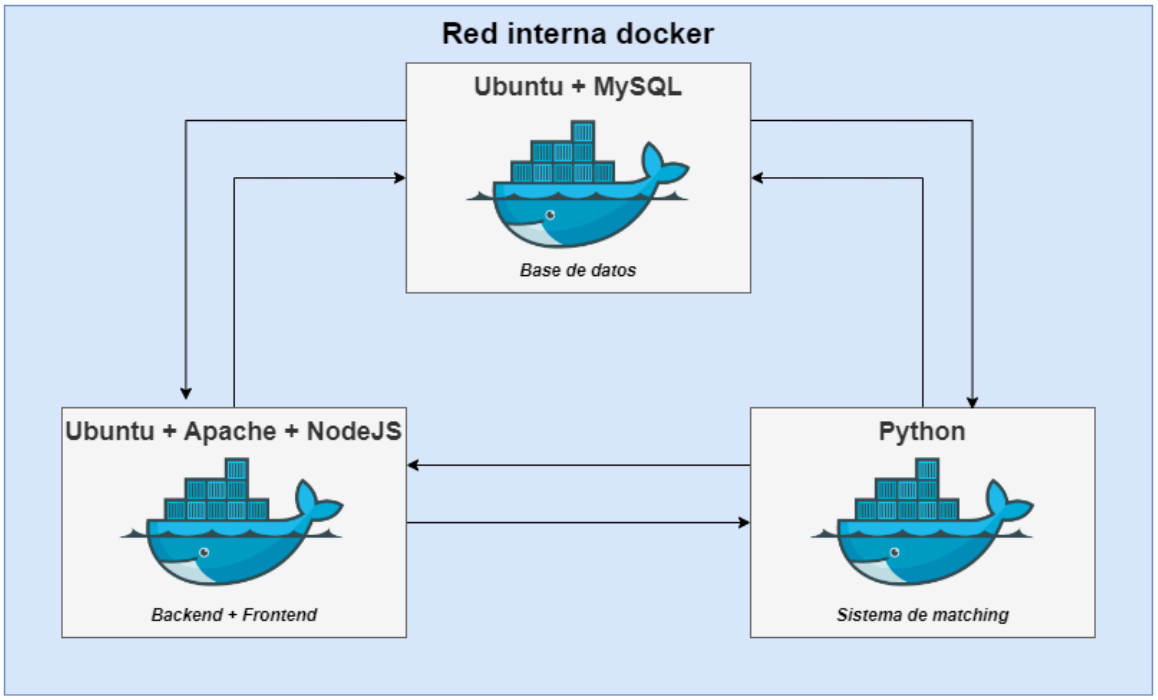

Figura 11. Arquitectura en Docker

La arquitectura construida en Docker consta de 3 contenedores y una red interna para la comunicación entre ellos:

Contenedor 1 (Ubuntu + Apache[65] + NodeJS): Este contenedor contiene la aplicación backend desarrollada en Node.js y también sirve los archivos estáticos del frontend. Para esto, se decidió instalar Ubuntu 22.04 con el servidor Apache y Node.js. A pesar del principio de Docker, que es el desacoplamiento y la modularización, se prefirió alojar el frontend y el backend en un solo contenedor para facilitar el manejo de las redirecciones entre el API y el frontend. Además, tanto el frontend como el backend comparten los certificados SSL generados, lo que se tuvo en cuenta al diseñar los

contenedores. En este contenedor, las peticiones son recibidas mediante el servidor de Apache y luego son distribuidas según corresponda. Mediante configuraciones de Proxy, como se mencionó anteriormente, si el dominio ingresado es https://puente.mdp.edu.ar/api, Apache hace una redirección a localhost:3000/api, utilizado por la aplicación de Node.js. Además, si la petición requiere una consulta al sistema de matching, el backend se encargará de realizar una petición HTTPS al contenedor que aloje el sistema de matching.

Contenedor 2 (Sistema de matching): Para este contenedor, se utilizó la imagen de Python 3.11.1, ya que era lo único necesario para su funcionamiento.

Contenedor 3 (Base de datos): Este contenedor contiene la base de datos. Para generar este contenedor, se utilizó la imagen de Ubuntu y se instaló MySQL para su posterior uso. Se podría haber utilizado directamente la imagen de MySQL, pero surgió el problema de que carecía del módulo "crontab", utilizado para ejecutar un script de backup de la base de datos. Se intentó instalar este módulo en el contenedor con la imagen de MySQL, pero luego de investigar, se descubrió que la imagen de MySQL es muy ligera y optimizada, por lo que no permite instalar módulos como "crontab". Por este motivo, se utilizó Ubuntu y se generó el script para el backup de la base de datos, como se mencionó en la sección "Backups".

Red interna: La red interna de Docker se creó para garantizar la comunicación entre varios contenedores. Primero se creó la red y luego se agregaron los contenedores en el siguiente orden: Contenedor 3, Contenedor 2 y finalmente Contenedor 1. Esto se hizo así porque el Contenedor 1 necesita conocer la dirección del Contenedor 3 y del sistema de matching, mientras que el sistema de matching debe conocer la dirección del Contenedor 3. Por lo tanto, se generaron en ese orden, se investigaron las direcciones de los contenedores y se pasaron a los siguientes contenedores mediante variables de entorno.

Todos los contenedores fueron generados mediante sus respectivos Dockerfiles, los cuales se pueden encontrar en la sección "Anexo VI: Dockerfiles". Además de los Dockerfiles, se incluye un manual de despliegue de los contenedores, el cual será entregado a los responsables para que los técnicos de la Universidad puedan mover esos contenedores en el futuro si así lo desean. Este manual se encuentra en "Anexo V: Manual de despliegue". Esto se debe a que en algunos casos no es suficiente con generar los contenedores con los Dockerfiles, sino que también es necesario realizar configuraciones adicionales dentro del contenedor. Por este motivo, se dejó previsto el manual de despliegue.

4.15 - Certificados SSL

Un requisito importante era que la aplicación debía funcionar mediante HTTPS, no solo para evitar ataques que pueden ocasionar robo de información, sino también para que los usuarios no teman al ingresar a la aplicación debido a las advertencias de los navegadores ante los dominios sin certificado.

Para la generación de los certificados, como se mencionó anteriormente, se utilizó "certbot[66]" de la autoridad certificadora Let's Encrypt. Se optó por esta herramienta debido a que es gratuita y ampliamente utilizada en el mercado. La duración de los certificados es de 90 días.

En la documentación oficial de Let's Encrypt, se encuentra detallada la forma de instalar certbot y generar los certificados. Se siguieron esos pasos para la instalación, generando los certificados para el dominio "puente.mdp.edu.ar".

Además de la generación de los certificados, los mismos deben ser configurados en el servidor Apache. Para ello, dentro de los VirtualHost del servidor, se deben agregar las direcciones hacia los certificados.

Dado que los certificados deben renovarse, se optó por apuntar las configuraciones hacia enlaces simbólicos. Esto se hizo porque al momento del despliegue ya se contaba con certificados que no habían vencido, por lo que primero se configuraron esos y luego se preparó la renovación automática. Entonces, en el archivo de configuración de VirtualHost con SSL, se agregaron las siguientes líneas:

SSLCertificateFile /etc/ssl/certs/cert.pem
SSLCertificateKeyFile /etc/ssl/private/privkey.pem
SSLCertificateChainFile /etc/apache2/ssl.crt/chain.pem

Los archivos siguientes son los que “certbot” genera:

SSLCertificateFile: Apunta a cert.pem. Es el archivo que contiene el certificado digital emitido para tu dominio por una Autoridad Certificante. Este contiene nombre del dominio, clave pública y validez del certificado

SSLCertificateKeyFile: Apunta a privkey.pem. Este archivo es el que contiene la clave privada correspondiente al certificado.

SSLCertificateChainFile: Apunta a chain.pem Este archivo contiene el conjunto de certificados intermedios y también el certificado raíz de la Autoridad Certificante que emitió el certificado

Las líneas agregadas apuntan al archivo indicado, el cual a su vez apunta a la carpeta donde certbot genera los certificados. Esto significa que cada vez que se renueven, no será necesario realizar ninguna modificación.

Quizás se genere la siguiente pregunta, ¿por qué no se apuntó directamente la dirección a donde certbot genera los certificados en lugar de apuntar a un enlace simbólico que apunte a este último? La respuesta radica en que certbot solo permite generar certificados en un servidor que esté efectivamente apuntado por el dominio ingresado, en este caso "puente.mdp.edu.ar". El problema surgió porque mientras se estaba desarrollando en el entorno de desarrollo, ningún servidor con ese dominio estaba apuntando, ya que estaba en el entorno de desarrollo. Por lo tanto, se utilizaron los certificados que aún eran válidos y se instalaron hasta que expiraron. Luego, cuando la aplicación esté alojada en el servidor apuntado por el dominio mencionado, se ejecutará el comando para la renovación automática del certificado. Se optó por este enfoque para evitar que la aplicación estuviera un día sin certificado mientras se realizaban las configuraciones necesarias.

Para la ejecución del script para la renovación del certificado cada 90 días, se utilizó nuevamente "crontab". Para ello, se generó un script que contenía únicamente el comando "certbot renew". Con eso solo, fue suficiente para que el certificado se renovara. La línea ingresada en "crontab" fue la siguiente:

*0 0 */90 * * certbot renew*

Luego, para la aplicación de Node.js también se necesitó instalar el certificado. Esto se debe a que el ProxyPass también debe funcionar a través de HTTPS. En un principio, el flujo de la petición a la API era el siguiente:

https://puente.mdp.edu.ar/api/recurso

Esto llegaba al servidor de Apache cifrado mediante HTTPS. Luego, con el ProxyPass, se enviaba hacia:

http://localhost:3000/api/recurso

Esto significa que al servidor, la petición llegaba cifrada con HTTPS, pero del servidor hacia la aplicación de Node.js no lo estaba. Por este motivo, se instalaron los certificados para la aplicación de Node.js y ahora el ProxyPass se envía hacia:

https://localhost:3000/api

El cual también está cifrado por HTTPS.

Como ya se mencionó, en la sección "Anexo: Manual de despliegue" se puede encontrar información aún más precisa de cómo se configuraron los certificados.

5 - Producto

5.1 - Producto obtenido

El producto desarrollado es una aplicación web compuesta de una interfaz de usuario, un servicio de backend y un servicio de sistema de matcheo, junto con la base de datos correspondiente.

La interfaz de la aplicación o el frontend se comunicará con la API expuesta por el backend, proveyendo los recursos necesarios para poder funcionar correctamente.

El usuario, dependiendo los roles, puede acceder a varias interfaces, en detalle son:

Usuario sin loguear:

Pantalla de inicio de sesión: Permite a un usuario iniciar sesión y dirigirse al registro y olvide mi contraseña.

Pantalla de registro: Puede registrar un usuario tanto como oferente o demandante.

Pantalla de home: Landing page de la plataforma. Los visitantes aterrizan aquí y contiene la información de uso de la plataforma.

Visualizar las ofertas: Visualización principal de las ofertas. Al scrollear hacia abajo, se desplegarán tarjetas de información de una oferta.

Más información de la oferta:

Visualizar las líneas de financiamiento: Se visualizará una tabla con todas las líneas de financiamiento disponibles. Al clickear en un icono de lupa, se redireccionará a la información detallada de la misma.

Más información de la línea:

Visualizar organizaciones participantes: Se muestra una página donde aparecen todos los logos de las empresas participantes en Puente.

Usuario logueado:

Menú principal: Menú principal dinámico de usuario. Aquí se pueden visualizar todas las acciones de un usuario. A medida que el usuario acumula roles, más opciones tendrá disponible.

Administrar mi usuario: Formulario donde un usuario puede modificar su nombre y apellido, y redirigir a cambiar mi contraseña.

Oferente:

Agregar oferta: Formulario de carga de oferta. El usuario oferente podrá ingresar todos los datos de la misma, y una vez finalizada la carga, la misma queda pendiente de autorización

Mis ofertas: Se visualizan todas las ofertas subidas por el usuario en forma de tarjetas de presentación. Se visualiza su estado e información reducida de la misma. Se cuentan con opciones para modificar, obtener más información o eliminar

Consultar demandas: Se muestra una tabla con todas las demandas disponibles en la web. La tabla contiene el icono de una lupa para ver más información de la demanda.

Más información de la demanda: Se visualiza la demanda con toda la informacion completa en forma de formulario

Mis matches: Se muestra una tabla con todas las ofertas del usuario, una breve descripción, una cantidad de matches, y un botón para ampliar información del match

Demandante:

Agregar demanda: Formulario de carga de demanda. El usuario demandante podrá ingresar todos los datos de la misma, y una ves finalizada la carga, la misma queda en estado pública

Mis demandas: Se visualizan todas las demandas subidas por el usuario en forma de tabla. Se visualiza su estado e información resumida de la misma. Se cuentan con opciones para modificar, obtener más información o eliminar.

Consultar ofertas: Visualización principal de las ofertas. Al scrollear hacia abajo, se desplegarán tarjetas de información de una oferta. Contiene un botón para visualizar más información de la misma

Mis matches: Se muestra una tabla con todas las demandas del usuario, una breve descripción, una cantidad de matches, y un botón para ampliar información del match.

Administrador Unidad Oferente:

Ofertas nivel 1: El usuario podrá ver las ofertas que estén todavía en estado pendientes y pertenezcan a la unidad académica a la cual esté vinculado el usuario, pudiendo autorizarlas o denegarlas explicando el motivo. Las ofertas se visualizan también en formato de tarjetas.

Agregar líneas de financiamiento: Este rol podrá ingresar una línea de financiamiento nueva, completando el formulario correspondiente.

Consultar líneas de financiamiento: El usuario podrá consultar todas las líneas de financiamiento disponibles, sin importar su estado.

Administrador Rectorado:

Administrar unidad oferente: Se mostrará una tabla con todas las unidades oferentes disponibles. Clickeando en el icono de la lupa, se mostrará más información y se podrá modificar la misma. Clickeando en el icono del cesto de basura, se podrá eliminar la unidad oferente. Además, clickeando en “Agregar unidad oferente” se redireccionará a un formulario para cargar una nueva unidad oferente.

Administrar tipos unidad oferente: Se mostrará una tabla con todos los tipos de unidades oferentes disponibles. Clickeando en el icono de la lupa, se mostrará más información y se podrá modificar la misma. Clickeando en el icono del cesto de basura, se podrá eliminar el tipo de la unidad oferente. Además, clickeando en “Agregar tipo de unidad oferente” se redireccionará a un formulario para cargar un nuevo tipo de nueva unidad oferente.

Administrar unidad académica: Se mostrará una tabla con todas las unidades académicas disponibles. Clickeando en el icono de la lupa, se mostrará más información y se podrá modificar la misma. Clickeando en el icono del cesto de basura, se podrá eliminar la unidad académica. Además, clickeando en “Agregar unidad

académica" se redireccionará a un formulario para cargar una nueva unidad académica.

Ofertas nivel 1: El usuario podrá ver las ofertas que estén todavía en estado pendientes y pertenezcan a la unidad académica a la cual esté vinculado el usuario, pudiendo solamente visualizarlas. Las ofertas se visualizan también en formato de tarjetas.

Ofertas nivel 2: El usuario podrá ver las ofertas que estén todavía en estado autorizadas y pertenezcan a la unidad académica a la cual esté vinculado el usuario, pudiendo autorizarlas o denegarlas explicando el motivo. Las ofertas se visualizan también en formato de tarjetas.

Consultar contactos: El administrador rectorado podrá visualizar en forma de tabla, todas las ofertas (no solo autorizadas) y la cantidad de contactos que han tenido. Clickeando en el icono de lupa, se desplegará otra tabla donde se verán todas las demandas que contactaron a esa oferta.

Consultar matches de oferta: En esta sección, el administrador podrá visualizar todas las ofertas, y su cantidad de matches. Además clickeando en el icono de la lupa, se podrán visualizar todas las demandas que hayan matcheado con la demanda seleccionada.

Consultar matches de demanda: En esta sección, el administrador podrá visualizar todas las demandas, y su cantidad de matches. Además clickeando en el icono de la lupa, se podrán visualizar todas las ofertas que hayan matcheado con la demanda seleccionada

Consultar administradores de unidades académicas: Aquí se generará una tabla que contiene todos los administradores de las diferentes unidades académicas.

Administrador General:

Administrar usuario: El administrador general podrá hacer modificaciones sobre los otros usuarios. Se mostrará una tabla con

todos los usuarios existentes. Podrá dar de baja un usuario dándole al icono de un cesto de basura, o ampliar información clickeando en el icono de la lupa.

Detalle usuario: Se mostrará la información detallada del usuario, donde el administrador podrá cambiar los roles de ese usuario.

5.2 - Rendimiento (Google Insights)

A la aplicación se le realizaron pruebas de rendimiento para evaluar su funcionamiento. Para este propósito, se empleó Google PageSpeed Insights[67], una herramienta que proporciona información detallada sobre los tiempos de carga de la página y ofrece sugerencias para mejorar aspectos como SEO, rendimiento, entre otros. En la fase inicial de las pruebas, estos fueron los resultados obtenidos:

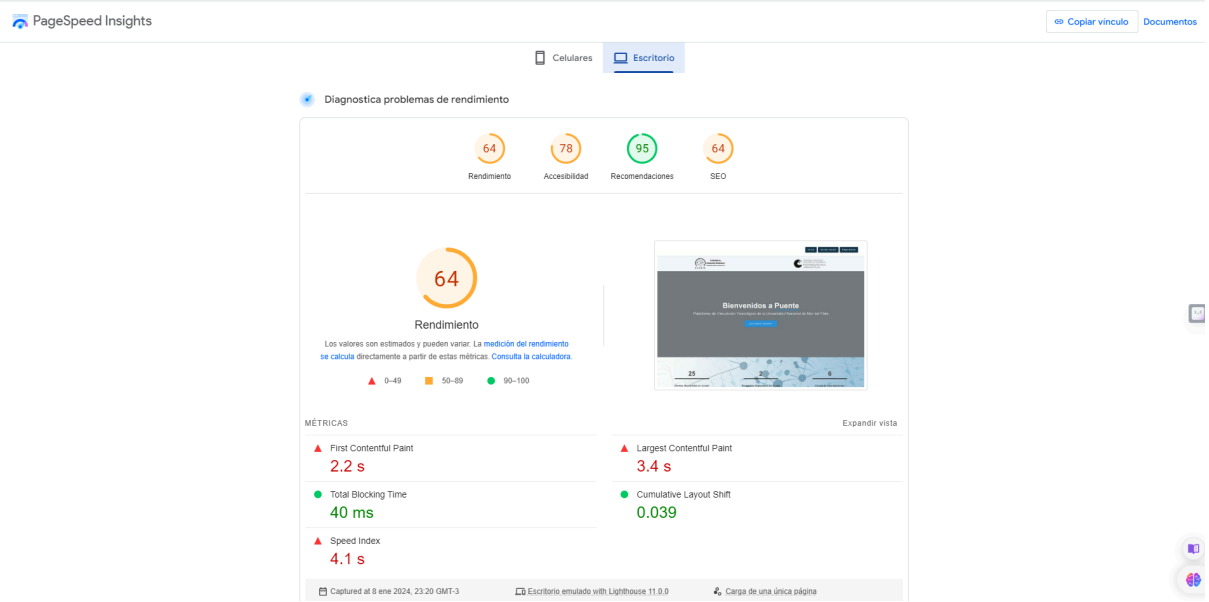

Figura 12. Test de rendimiento y SEO en Google PageSpeed Insights

Posteriormente, se mejoró lógicamente el rendimiento de la aplicación implementando las mejoras recomendadas. Estas incluyeron la optimización de imágenes, la eliminación de código no utilizado, entre otras acciones. Los resultados actualizados de las pruebas de rendimiento fueron los siguientes:

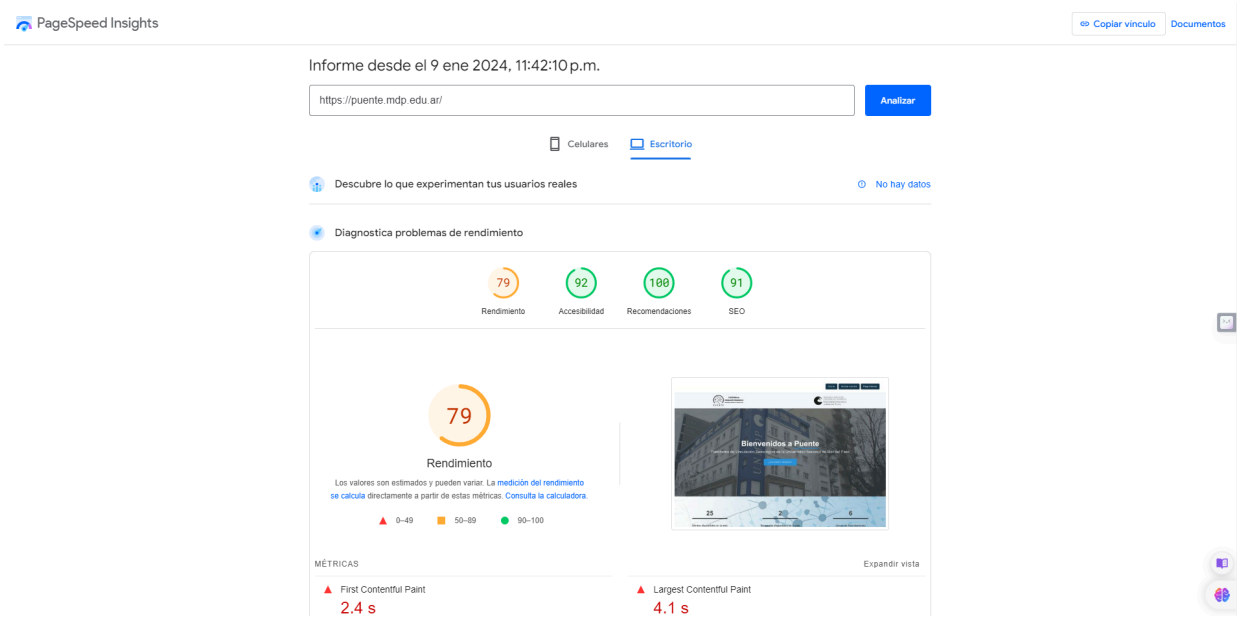

Figura 13. Test de rendimiento y SEO en Google PageSpeed Insights posterior

Como se puede observar, el rendimiento general llegó a 79 puntos, mientras que se mejoraron otros aspectos notoriamente, como es el SEO y la accesibilidad.

5.3 - Trabajos futuros

Actualmente, contamos con un software que satisface todas las necesidades establecidas por la Secretaría de Vinculación Tecnológica. Sin embargo, se reconoce que el alcance de este proyecto ha sido limitado y que existen diversas mejoras que pueden implementarse en el futuro.

Una de las mejoras principales podría centrarse en el perfeccionamiento de la red neuronal. Este es un proyecto de gran envergadura que requiere recursos de hardware para entrenar la red, recursos que el desarrollador actual no posee en la actualidad. Además, se necesita un conocimiento adicional en el afinamiento de redes neuronales para optimizar su rendimiento.

Otro aspecto importante para futuros trabajos podría ser el desarrollo de una aplicación móvil descargable disponible en las principales tiendas de aplicaciones, como Android e IOS. Esto proporcionaría una experiencia más cómoda para los usuarios, ya que trabajarían desde una aplicación nativa en lugar de acceder a través de un navegador web.

El rediseño visual de la aplicación también podría ser una área de mejora significativa. Dado que los conceptos de diseño y experiencia de usuario suelen ser manejados por diseñadores UX/UI especializados, es posible que haya aspectos que puedan mejorarse, especialmente teniendo en cuenta que el desarrollo de Puente fue realizado únicamente por un estudiante.

La base de datos no fue sometida a un examen de estrés desde un principio, debido a que por la envergadura del proyecto no se consideró necesario. Como trabajo futuro, sobre todo si la aplicación escala exponencialmente, incrementado el número de entradas de la base de datos, es recomendable hacer un test de estrés para analizar si se deben hacer modificaciones a la misma.

Una tarea que quedó fuera del alcance de este proyecto fue el desarrollo de una métrica de calidad para el sistema de matcheo. Por lo tanto, se sugiere considerar este aspecto como un punto fundamental en futuros trabajos, especialmente si se busca realizar un análisis exhaustivo del rendimiento del sistema. Este análisis cobra aún más relevancia en caso de que el sistema de matcheo experimente un crecimiento significativo.

Por último, aunque los requisitos actuales se centran en la parte de vinculación, la plataforma no aborda aspectos burocráticos y la gestión de pagos. Sería interesante expandir la funcionalidad de la aplicación para cubrir estos aspectos y garantizar que sea lo más completa posible para satisfacer las necesidades de la Secretaría de Vinculación y Transferencia.

6 - Presentaciones y capacitaciones

Llegado a una etapa crucial en el desarrollo de la plataforma, se iniciaron las capacitaciones y presentaciones. El primer paso fue evaluar la idoneidad de la plataforma para los usuarios, considerando cualquier cambio necesario en respuesta a sus necesidades. Dado que Puente fue diseñada para resolver problemas específicos de los desarrolladores, era imperativo obtener sus opiniones.

Este proceso se basó en visitas a diversas facultades, donde se presentó la plataforma a los directores de transferencia de cada unidad académica. Se les demostró cómo cargar información, se resolvieron dudas y se recopilaron sus valiosas recomendaciones.

Además, se facilitó a los usuarios el contacto directo con el desarrollador a través de correo electrónico y número de celular, permitiéndoles comunicar cualquier pregunta o sugerencia. Los usuarios también tenían la opción de solicitar ser administradores de unidad oferente, para lo cual se comunicaban con Manuel Conde o el desarrollador, y se les asignaba el rol correspondiente.

Las capacitaciones se llevaron a cabo en las facultades de Humanidades, Ingeniería y Derecho, así como en reuniones masivas en el aula magna del rectorado de la Universidad.

Las capacitaciones no fueron las únicas interacciones presenciales. Se realizaron diversas presentaciones de la aplicación ante una audiencia diversa, con el objetivo de dar visibilidad a Puente tanto entre docentes de diferentes áreas como entre demandantes de diversos sectores. Este esfuerzo se complementó con la promoción en las redes de la Universidad y la Secretaría de Vinculación, que regularmente compartían flyers sobre Puente.

Puente fue destacada en eventos como INTECMAR, donde diversas facultades y desarrolladores presentaron sus proyectos en busca de financiamiento. Posteriormente, en el INTEMA, se realizaron reuniones de presentación de Puente, donde los secretarios de Vinculación aprendieron sobre el uso básico de la plataforma. La última presentación tuvo lugar en el evento del instituto de electrónica ICYTE, donde se mostró un ejemplo práctico de la aplicación para todos los asistentes. Dentro de la sección Anexo VII: Imágenes de presentaciones se podrán encontrar todas las imágenes de las distintas presentaciones o capacitaciones.

Además de capacitaciones presenciales, también se le solicitó al autor un video de capacitación de usuarios, donde se explicó cómo cargar una nueva oferta a la plataforma en un seguimiento paso a paso, mostrando errores frecuentes y consejos a la hora de usar la plataforma. Este video se puede encontrar en la sección Anexo VIII: Video de Capacitación.

6.1 - Resultados de las reuniones

Tanto las presentaciones como las reuniones que se hicieron durante el desarrollo de Puente, tuvieron como objetivo poder compartir el funcionamiento de la plataforma a los usuarios y a los demandantes, y poder así encontrar errores o recomendaciones para poder entregar un software de calidad. Teóricamente y en la mayor parte de los casos, las

reuniones clarifican el camino y dejan tareas concretas por hacer, ya sea modificar el software presentado o agregar nuevas funcionalidades al mismo.

Particularmente, existieron reuniones que no fueron tan satisfactorias en un principio, y que dejó un período de incertidumbre en cómo seguir encarando el desarrollo del proyecto.

El ejemplo perfecto para esta problemática, fue la reunión concretada el día 7 de septiembre de 2023, en sala de reuniones del edificio de rectorado de la Universidad Nacional de Mar del Plata, con los distintos secretarios de vinculación de las distintas unidades académicas. En total, la asistencia fue de aproximadamente 20 personas. Junto con Fernando Genin, se mostró la interfaz de la plataforma en una versión primitiva, y el flujo principal de la carga de ofertas de la misma. Previamente, se pensó que esta reunión no sería de mucha extensión y que lo desarrollado hasta el momento era acorde a las preferencias de los secretarios. Sin embargo, un debate muy amplio fue desarrollado, con una duración aproximada de tres horas. Se discutieron temas de privacidad, flujo de carga de una oferta, campos que no debían faltar en las descripciones de las ofertas, cuestiones de seguridad de los usuarios y niveles de autorización de las ofertas. En general, la reunión se basó en discusiones entre los secretarios de vinculación, y consultas técnicas al desarrollador.

Se tomaron notas sobre las necesidades de cada usuario, pero desafortunadamente muchas se contradecían entre sí, sobre todo en los temas relacionados con el proceso de autorización de las ofertas, donde particularmente se focalizó en quienes serían los administradores. Muchas opiniones para mejorar la plataforma, ocurrieron debido a que la idea principal fue pensada solamente por Guillermo Lombera, quien conocía mayormente las necesidades, pero evidentemente no en su totalidad. Es por eso, que varios secretarios plantearon diversos temas para agregar o modificar. Entre ellos, los más relevantes fueron:

- **Privacidad de la plataforma:** Aquí se manifestó principalmente sobre cómo los datos de usuarios serían almacenados y quienes podrían crearse una cuenta con rol de oferente.
- **Autorizaciones de las ofertas:** Donde se planteó un cómo las ofertas debían ser aprobadas en caso que cumplieran los requisitos.
- **Visibilidad de ofertas y demandas:** Se debatió si las ofertas y demandas debían ser públicas para cualquier visitante en caso de estar aprobadas. Esto generó discusiones entre secretarios, y se tuvo que volver a pensar cómo cumplir todos los requerimientos solicitados en este apartado.
- **Campos de carga:** Se trató de que campos de información eran vitales a la hora de cargar una oferta. Luego de esta reunión, se agregó el campo de “FAQS” (Preguntas

frecuentes), donde los usuarios oferentes pueden cargar información extra sobre la oferta que se esté cargando.

Una vez finalizada la reunión, se debió recopilar toda la información presentada en la reunión, para luego poder detallar los pasos a seguir en el desarrollo de Puente. Para sorpresa del desarrollador, se creó un período de incertidumbre donde no estaba claro el rumbo a seguir, producto de las distintas opiniones de los Secretarios.

A pesar de estas dificultades y gracias a semanas de trabajo, se pudieron sobrepasar las problemáticas y encontrar un punto en todos los nuevos requerimientos generados por los secretarios, para así continuar con el desarrollo. Esto dejó una aprendizaje muy grande al desarrollador, quien se encontró con el primer gran problema en el mundo del desarrollo, que es la incertidumbre y el miedo de no poder lograr lo que se está pidiendo.

Por fortuna, el resto de las reuniones logradas fueron productivas y dejaron un rumbo claro en lo que los usuarios requerían, y colaboraron de una forma amena con el desarrollo. En la sección Anexo VII: Imágenes de presentación se puede observar una imágen de la reunión con todos los Secretarios de Vinculación.

7 - Memoria del proyecto

7.1 - Planificación esperada vs ejecución

Como se destacó anteriormente en la sección de planificación inicial, el desarrollador inicialmente carecía de la experiencia necesaria en la planificación, lo que condujo a ciertas redefiniciones en el cronograma.

En un principio, se adoptó un esquema en cascada, que implicaba considerar las etapas de análisis de requerimientos, conformidad del usuario, desarrollo, testing, documentación, implementación y capacitación de forma lineal, es decir, una etapa comenzaba cuando la anterior finalizaba sin posibilidad de retroceso. Este enfoque resultó ser un error derivado de la inexperiencia, ya que para obtener la conformidad del usuario, resultaba más eficiente elaborar diferentes módulos y probarlos. A pesar de esta corrección, los tiempos iniciales y finales del proyecto se estimaron correctamente.

Si bien no se utilizó un sistema de *checkpoints* para corroborar que se estaban cumpliendo los hitos del proyecto, cada dos semanas el desarrollador se comunicaba vía Google Meet con el tutor, donde se analizaba si el software estaba atrasado o no (como se detalla en la sección comunicaciones). Esto quiere decir que en cierta medida, cada dos semanas se corroboraba los avances planteados en las tomas de requerimientos anteriores, y se corroboraron las correcciones vistas en la anterior reunión. Más allá de eso, se detalla en las conclusiones, que esto fue un error en la estimación, ya que hubiese sido útil la utilización de checkpoints para conocer el estado del desarrollo.

La estimación inicial de 2008 horas totales fue acertada y proporcionó un tiempo adecuado para llevar a cabo el proyecto. Por lo tanto, a pesar de la replanificación, el proyecto no experimentó retrasos significativos.

En las etapas de documentación, implementación y toma de requerimientos, se mantuvo la metodología cascada, ya que se desarrollaron de manera lineal. Sin embargo, para el testing, desarrollo y conformidad del usuario, se optó por una metodología iterativa, como se detalla a continuación.

El nuevo Gantt de planificación, más detallado, resultó ser el siguiente:

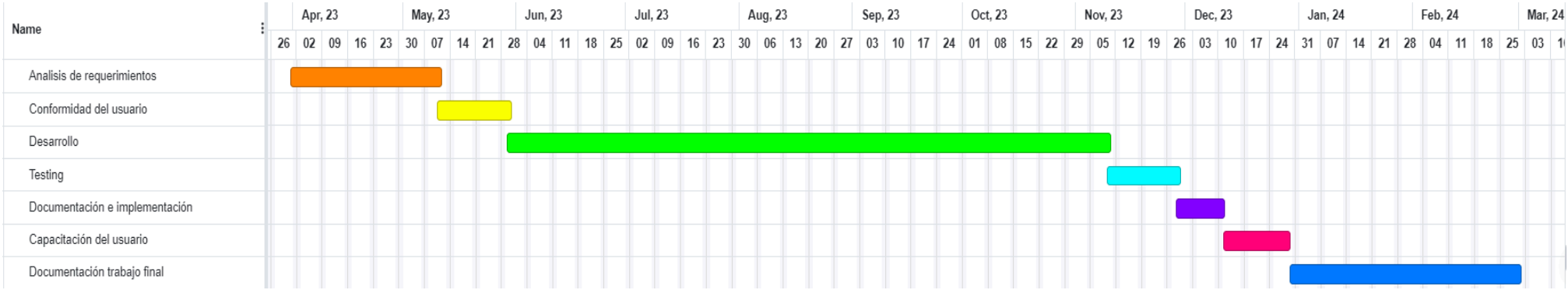

Figura 14. Diagrama de Gantt planificado

		abr-23				may-23				jun-23				jul-23				ago-23				sep-23				oct-23				nov-23				dic-23				ene-24				feb-24			
ID	Tarea	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
1	Análisis de requerimientos	30																																										
2	Conformidad del usuario					7							7					7					7					7					7												
2	Desarrollo						120																																						
3	Testing											7					7					7					7					7		7											
4	Documentación e implementación																																	15											
5	Capacitación del usuario																																			15								
6	Documentación del trabajo final																																					60							

Figura 15. Diagrama de Gantt ejecutado

Analizando el Gantt, podemos ver que la planificacion inicial fue sin considerar el solapamiento de las tareas, de forma lineal y en cascada. Luego en la práctica, se corrigió esto y se comenzaron a solapar etapas y a trabajar de forma iterativa. Por suerte, esto no causó desvíos en los tiempos de entrega, pero si logró que el desarrollador obtenga la experiencia suficiente para no cometer los mismos errores.

Como se observa, la fecha de inicio fue el primero de abril, y la de finalización el primero de marzo, siendo un total de 2008 horas. Esta estimación inicial, fue calculando un trabajo de 8 horas diarias por el total de los días que se ocuparían en hacer el proyecto, lo cual se detallará en las conclusiones como un error.

7.2 - Análisis de las etapas

7.2.1 - Análisis de requerimientos

La primera fase del proyecto, centrada en el análisis de requerimientos, se destacó por su estabilidad y ausencia de modificaciones significativas. Este proceso implicó la traducción de las ideas proporcionadas por Guillermo Lombera a un diseño de software concreto. Cumpliendo con la planificación inicial, se completó en un periodo de 30 días durante los cuales se recopilaron los requerimientos y se elaboró un documento detallado que describía las características esenciales que debía tener la plataforma.

En este caso, la metodología seguida podría considerarse como en cascada, ya que no se procedió a la siguiente etapa hasta que la anterior estuvo completamente concluida. La rigidez de esta metodología garantizó que cada fase se cerrará de manera integral antes de avanzar, evitando la necesidad de revisiones o tomas adicionales de requerimientos.

Una particularidad destacada fue la ausencia de una segunda toma de requerimientos, ya que las etapas se finalizaban y eran validadas por el referente funcional, como se visualiza en el diagrama de Gantt. Esta validación continua durante el proceso permitió asegurar que los requisitos iniciales se mantuvieran consistentes y alineados con la visión del proyecto.

En resumen, la fase de análisis de requerimientos se ejecutó de manera fluida y eficiente, siguiendo una metodología en cascada que proporcionó estabilidad y claridad en el progreso del proyecto. Este enfoque demostró ser efectivo en situaciones donde la definición temprana y precisa de los requisitos es esencial para el éxito del proyecto.

7.2.2 - Conformidad del usuario

En esta fase del proyecto, se observan cambios más significativos en la estructura y enfoque del trabajo. Se tomó la decisión de dividir y prolongar la etapa, considerando la posibilidad de trabajar de manera colaborativa con el equipo de desarrollo para optimizar el tiempo, evitando los desperdicios que se habían identificado en la planificación anterior.

En este nuevo enfoque, cada vez que se completaba un módulo, se validaba de inmediato con el referente funcional para obtener la aprobación y poder avanzar según la planificación. En caso de que el referente identificara errores o mejoras necesarias, se requería corregirlos antes de avanzar. Esta validación continua permitió mantener un flujo eficiente y minimizar la posibilidad de desviaciones importantes en el proyecto. La duración de esta tarea se extendió a 6 semanas, asignando una semana adicional para cada módulo entregado. Esta prolongación se justificó para coordinar eficazmente las reuniones de validación y abordar las correcciones de manera efectiva.

Este cambio en la duración refleja una adaptación hacia una metodología más iterativa, que se alinea con la tarea de desarrollo y testing. La inclusión de tiempo adicional no solo facilita la coordinación de correcciones, sino que también promueve una mejora continua basada en la retroalimentación constante del referente funcional. Esta fase, marcada por una mayor flexibilidad y colaboración, demuestra la evolución hacia prácticas más ágiles y eficientes en la gestión del proyecto de software.

7.3.3 - Desarrollo

En la fase de desarrollo del proyecto de software, se asignaron las horas planificadas con precisión para llevar a cabo las tareas programadas, la duración fue de 960 horas. La estrategia adoptada consistió en organizar el desarrollo de manera modular, permitiendo una ejecución más eficiente y prolija. La división por módulos facilitó una aproximación iterativa, permitiendo que las etapas de desarrollo, testing y conformidad del usuario se superpusieran de manera coordinada.

Sin embargo, un aspecto a tener en cuenta para futuros proyectos es el enfoque lineal adoptado en la parte de diseño de los módulos. En lugar de abordar el diseño de manera previa y estructurada, se optó por un enfoque más flexible, agregando o eliminando elementos según las necesidades surgidas durante el desarrollo. Aunque este método permitió una adaptación ágil, se considera una mala práctica debido a la falta de un

horizonte claro en la fase de diseño. A pesar de este desafío, se logró llevar a cabo todo de manera óptima, demostrando la flexibilidad del equipo de desarrollo.

Es relevante señalar que la etapa de desarrollo fue la de mayor duración, lo cual es una ocurrencia común en proyectos de software. Esta duración extendida se atribuye a la concentración de esfuerzos y a la inevitabilidad de enfrentar problemas inherentes a un desarrollo estándar. Además, la inexperiencia puede generar desafíos adicionales. Es crucial aprender de estas experiencias para mejorar las prácticas de desarrollo en futuros proyectos y así aumentar la eficiencia y la calidad del resultado final.

Inicialmente no se tenía noción de cuáles eran las tareas que se debían desarrollar, por lo que la estimación de la etapa de desarrollo fue difícil y errónea. Se consideraron 960 horas de desarrollo, tomando el desarrollo como algo único y no dividiéndolo en subetapas, por lo que no se puede analizar con exactitud los desvíos para cada una de las etapas. Finalmente las etapas (correspondiente a la confección de un módulo), con sus duraciones aproximadas, terminaron siendo:

- **Módulo usuarios [80 horas]:** El módulo de usuarios no ocupó mucho tiempo debido a que el desarrollador ya había desarrollado módulos de usuarios para otras plataformas, y conocía las tecnologías.
- **Módulo oferente [200 horas]:** El módulo de oferentes fue el más complicado (sacando el sistema de matcheo) debido a que fue el primero que se confeccionó, y al que más importancia se le dio debido a que sería el accedido por los docentes de la universidad. Además, se debió investigar sobre la estructura interna de unidades oferentes de la universidad, que demoró la confección de este módulo.
- **Módulo demandante [120 horas]:** Una vez confeccionado el módulo de oferentes, ya se tenía una base sólida para confeccionar el módulo de demandantes. Por lo tanto, no presentó mayores dificultades y por eso la duración es menor.
- **Módulo de administradores [160 horas]:** Este módulo abarcó todo lo que fue el sistema de roles y los anillos de autorizaciones de ofertas. En un principio involucró un gran desafío para el autor, debido a que se requería que el sistema de autorizaciones sea seguro y fácil de utilizar.
- **Módulo anónimo [80 horas]:** El módulo anónimo involucra todo lo que un usuario público puede hacer. Por lo tanto, este fue el módulo más sencillo de confeccionar.

- **Módulo de inteligencia artificial [320 horas]:** Equivalente al final del desarrollo, el módulo de inteligencia artificial fue el proceso que más tiempo tomó al ser un tópico el cual se debió investigar nuevas tecnologías desconocidas para el autor.

Un gran problema que se detallará en las conclusiones, es sobre el error que fue el no haber llevado una bitácora de las tareas que se realizaban día a día. Esto logró que no se sepa exactamente cuánto duró cada sub-etapa del desarrollo, y no tener un costo exacto de la aplicación.

7.3.4 - Testing

La etapa de testing se dividió en seis bloques correspondientes a los módulos desarrollados. De manera iterativa, se completaba un desarrollo, se realizaba el testing y luego se presentaba al usuario. En total, se dedicaron 336 horas de trabajo. Como sucedió con la conformidad del usuario, la planificación inicial no contempló el solapamiento de las tareas, lo que resultaba en un orden lineal de las tareas y pérdida de tiempo. Además, al no superponerse con otras tareas, no se podía dedicar más tiempo a la tarea, lo que provocaría retrasos en el proyecto.

En el último bloque de testing, se llevó a cabo una prueba de integración, es decir, la ejecución de todos los módulos juntos, y se realizaron pruebas repetitivas. Aunque este bloque consumió el tiempo asignado por completo, se observó un excedente significativo, ya que la mayoría de las pruebas ya se habían realizado previamente para cada módulo por separado.

7.3.5 - Documentación e implementación

Al concluir las etapas anteriores, se procedió a cerrar el proyecto. Esta fase, en comparación con la estimación inicial, se extendió cuatro días más, totalizando catorce días o 112 horas. La duración adicional se debió a ciertos problemas derivados de la inexperiencia del usuario en el manejo del despliegue en entornos específicos, lo que llevó a la necesidad de días adicionales de investigación.

Una vez desplegada la aplicación, se procedió a documentar el código. Esta tarea resultó relativamente sencilla, ya que el desarrollador conocía y recordaba perfectamente el

código fuente de la aplicación. Aunque la documentación fue repetitiva y, en algunos casos, tediosa, se contó con herramientas familiares como SwaggerJS, lo que permitió que la documentación del código fuera más dinámica y proporciona una interfaz de usuario para su visualización.

7.3.6 - Capacitación del usuario

La penúltima tarea fue la capacitación, la cual se llevó a cabo de manera lineal una vez finalizadas las etapas anteriores. Esta tarea no experimentó cambios, ya que siempre fue independiente de las otras etapas y se mantuvo según lo planificado. Aunque inicialmente pueda parecer poco tiempo asignado para la capacitación, se tuvo en cuenta la buena disposición del desarrollador y la posibilidad de realizar consultas a través de WhatsApp y correo, lo que permitió acortar significativamente los tiempos. Además, la plataforma en sí es muy intuitiva, lo que facilita la exploración libre por parte de los usuarios. Naturalmente, la duración de algunas capacitaciones fue más extensa que otras, dependiendo de la disposición y facilidad del usuario. La duración fue de 112 horas

7.3.7 - Documentación del trabajo final

Finalmente, se procedió a la documentación del trabajo final, es decir, la elaboración del informe que abarca todos los aspectos del proyecto. Esta etapa no sufrió modificaciones, y la fecha límite para la finalización de esta documentación se mantuvo para el 1 de marzo de 2024, dando un total de 120 horas. Aunque podría considerarse una parte del desarrollo menos compleja que las etapas anteriores, la redacción de un informe detallado sobre todo lo desarrollado presenta sus propias complejidades. Es por eso que se estimó una duración de dos meses para esta etapa, ya que el autor prefirió asignar un tiempo más holgado. Afortunadamente, a medida que avanzaba el desarrollo, el desarrollador documentó diversas actividades realizadas día a día, lo que facilitó la posterior elaboración del documento.

7.3.8 - Resumen

En resumen, la estimación inicial resultó acertada desde la perspectiva del usuario, ya que los tiempos no sufrieron modificaciones y se lograron cumplir con los objetivos establecidos. No obstante, desde el punto de vista del desarrollador, se produjeron cambios

significativos, especialmente en las metodologías utilizadas para ciertas etapas como el desarrollo, testing y conformidad del usuario.

Como experiencia clave, se destaca la lección de no subestimar ciertas actividades y problemáticas, así como la importancia de adoptar enfoques más avanzados al seleccionar metodologías. Planificar de manera exhaustiva resulta fundamental para evaluar si la ejecución avanza correctamente o si es necesario realizar ajustes en el camino.

8 - Conclusiones

El proyecto Puente se presentó con la finalidad de diseñar una solución eficiente para un nicho poco explorado: la transferencia y vinculación de proyectos con demandantes interesados. Además, se buscaba aprovechar al máximo el potencial de la Universidad. En un contexto donde la tecnología es parte integral de nuestras vidas diarias, la decisión fue informatizar este sector, aprovechando el gran margen de mejora que tenía y transformándolo en una presencia destacada en el mercado. Se era consciente de que, en general, las personas no suelen esforzarse demasiado en la búsqueda de algo específico, y Puente facilitaría despertar el interés de los demandantes con un vistazo general. Hasta ahora, los objetivos establecidos por el referente funcional, Guillermo Lombera, se están cumpliendo satisfactoriamente.

El proyecto, aunque de magnitud considerable, fue un desafío que inicialmente parecía complicado, especialmente al contar solo con un desarrollador y un tiempo limitado. La falta de diversidad de opiniones en el desarrollo fue un aspecto negativo, pero la eficiencia en la coordinación fue positiva. Sin embargo, en balance, la falta de trabajo en equipo se considera un aspecto negativo. A pesar de las limitaciones, el proyecto se desarrolló correctamente y se respetaron los plazos de entrega.

Dentro del proyecto, se utilizaron herramientas aprendidas en la carrera y se aplicó la "mente ingenieril" para encontrar soluciones a problemas inéditos. Se aprendieron nuevas herramientas, pero la versatilidad adquirida durante los años de estudio facilitó su incorporación. Además, se experimentó con nuevas metodologías de trabajo y se adquirió experiencia en la gestión de conversaciones con clientes, especialmente en el asesoramiento y la negociación. Estas habilidades son esenciales para un ingeniero y solo pueden desarrollarse en proyectos de esta envergadura.

Como ya se mencionó en otro apartado, para lograr este desarrollo el autor tuvo que sobrepasar un gran número de problemáticas, atravesando momentos de incertidumbre donde incluso se vió comprometida de la continuación del proyecto, cuando existieron problemas en el diseño inicial de las interfaces, que logró desvíos en los primeros meses de desarrollo. Estos problemas trajeron consigo un gran aprendizaje tanto técnico como de gestión, permitiendo luego sobrepasar los problemas y continuar adelante con el desarrollo.

Además, esta experiencia es valiosa para el currículum de un ingeniero, al permitir iniciar un nuevo desafío laboral, y ser reconocido por altos cargos de la Universidad, que de otra forma quizás no hubiera sido posible.

Como se ha comentado a lo largo de este informe, se han reconocido errores que se han cometido a lo largo del desarrollo. Cada uno de estos errores colaboró para que hoy en día el autor entienda que no se debe hacer en el proceso de desarrollo, diseño y planificación de un software

En primera medida, se destaca que fue un error el no documentar las horas en el desarrollo del proyecto y tampoco aclarar qué se hizo en cada una. Siempre se plantea una estimación que, como su nombre indica, no es certera y quizás no describe cómo será el proceso de desarrollo en su totalidad. Pero es una hoja de ruta donde se pueden medir los desvíos que se tuvo en cada etapa. Dado que en este caso la estimación no fue buena, debido a que no se contemplaron eventos como vacaciones ni percances, entre otras cosas, no se tuvo una buena hoja de ruta a seguir, más allá de que el proyecto no registró atrasos (sí desvíos). Además, al no tener una bitácora donde se registran los sucesos del día a día, no se puede comparar con exactitud qué fue lo que ocurrió en los desvíos y tampoco se puede conocer con exactitud el costo de la aplicación. Cuando se costea un desarrollo, se analiza qué tareas se hicieron y la cantidad de tiempo que demoraron, dado que existen tareas que tienen un costo más elevado que otras, como es el caso del desarrollo de inteligencia artificial. Es correcto estimar mal a priori, pero lo importante es poder detectar estos errores a posteriori, lo que en este caso no se pudo lograr satisfactoriamente.

Tampoco se hizo uso de los checkpoints, que hubieran sido de gran ayuda para tener en cuenta cuanto nos estábamos desviando en cada etapa. Más allá de esto, sí se mantenía una corroboración cada dos semanas mediante las llamadas vía Google Meet, como se comentó en la sección planificado vs ejecutado.

Todo esto antes mencionado se reconoce como una debilidad en la gestión que no debe volver a ocurrir en desarrollos futuros. Particularmente, el no llevar una bitácora afectó a este trabajo debido a que no se puede conocer el costo con exactitud, ni tampoco se puede utilizar la estimación para casos futuros debido a que es muy inexacta.

También se puede comentar un aprendizaje relevante que se obtuvo, que es el hecho de no obviar en pedir los requerimientos técnicos para el desarrollo desde un principio. Como se detalló en el informe, existieron problemas sobre donde se debía alojar la aplicación y sobre las limitaciones técnicas que tenían los servidores de la Universidad. El desarrollador presentó un análisis de qué tecnologías eran convenientes para encarar este proyecto, pero la realidad es que terminaron siendo un estorbo para la Universidad, debido a que se tuvo que plantear una capa más a la solución (la utilización de Docker), cuando si se hubiera analizado los requerimientos técnicos en un principio, quizás se podrían haber usados otras tecnologías (como PHP) y estos problemas no hubieran existido. Este error, pudo haber costado la implementación del proyecto ya que, si no se encontraba una solución, Puente no podría estar disponible en los servidores de la Universidad. A pesar de esto, el autor se quita parte de la responsabilidad, debido a que el demandante no presentó que existían regulaciones técnicas para esto, por lo tanto procedió a utilizar las tecnologías con las que más experiencia tenía.

Ya habiendo terminado el proyecto, se considera que fue un error el no haber contado con un compañero de equipo. Distintas circunstancias, como los requerimientos técnicos amplios del proyecto, o la falta de compañeros disponibles para realizar el proyecto final, llevaron a que un solo integrante realice este trabajo. En reiteradas ocasiones, se presentaron situaciones en las que el autor necesitaba debatir con otra persona sobre qué decisión tomar, cómo resolver un problema, o el simple hecho de dividir las tareas entre dos o más personas. El no tener compañero de proyecto logró que en varias situaciones el autor experimente estrés y cansancio, lo que logró en cierta medida disminuir la productividad del mismo. En experiencias futuras, claramente la tendencia será formar un equipo de trabajo.

Este proyecto le dejó muchos aprendizajes al autor. Se deja en evidencia que se cometieron una gran cantidad de errores, pero se destaca la importancia de utilizarlos para que no vuelvan a ser cometidos en un futuro. Sin lugar a dudas, el aprendizaje y los conocimientos adquiridos son el punto más positivo que el desarrollador se lleva de este proyecto, dejando un saldo totalmente favorable.

Los objetivos planteados para este proyecto fueron cumplidos, llegando a obtener un software funcional, utilizando tecnologías como Angular, NodeJS, y MySQL para el desarrollo de la aplicación, otras como Python (con distintos modelos de IA, como Transformers) para el desarrollo del sistema de matcheo, y otras complementarias, como Docker y GIT para el despliegue de la aplicación final. Se evidencia que se logró potenciar la conexión entre los docentes e investigadores universitarios con el sector socioproductivo de la ciudad. Además, se logró que la Secretaría de Vinculación y Transferencia Tecnológica sea más eficiente, ya que actualmente es consciente de la cantidad de ofertas y demandas activas al momento, cosa que antes de la existencia de Puente, le era totalmente imposible. En la sección producto obtenido, se puede observar el detalle del software desarrollado. El software satisface a los demandantes, ya que han manifestado su conformidad con el mismo y los números de ofertas y demandas se incrementan día a día, mostrando también una aceptación por parte de los usuarios. Luego, además de los objetivos del proyecto, también se cumplieron objetivos personales del desarrollador, como, por ejemplo, poder lograr gestionar un proyecto a mediana escala sin equipo de desarrolladores, aprender tecnologías complejas como las que se utilizaron para el desarrollo de la inteligencia artificial y, más importante, poder sobrepasar momentos de incertidumbre, donde incluso el desarrollo del proyecto corrió riesgo.

Para concluir, el desarrollo se llevó a cabo con éxito en condiciones desafiantes. Puente simplificará significativamente la tarea de conectar a investigadores con demandantes, beneficiando a cientos de investigadores y miles de demandantes. Personalmente, la experiencia fue satisfactoria y enriquecedora, contribuyendo considerablemente al crecimiento de conocimientos y habilidades.

9 - Anexos

Anexo I - Documento contra ataques

Documento formato “xlsx” adjuntado en la entrega. El nombre del documento es: “Documentacion_contra_ataques.xlsx”

Anexo II - Documentación API REST

Dentro de la carpeta “Documentación API Puente”, abrir con el navegador el archivo en formato HTML, “Swagger UI”, donde se abrirá la interfaz de la documentación de Puente.

Anexo III - Diagrama Entidad-Relación.

Archivo en formato “pdf” el diagrama Entidad-Relación de la base de datos de Puente. El nombre del documento es: “Diagrama_Entidad_Relacion.pdf”

Anexo IV - Script de backup de la base de datos

Archivo en formato “sh” donde se encuentra el script que se ejecutará para el backup de la base de datos. El mismo puede ser ejecutado con cualquier editor de texto y ver su contenido. El nombre del documento es: “Backup.sh”

Anexo V - Manual de instalación de Puente con docker.

Archivo en formato “pdf” donde se puede ver los distintos caminos para el despliegue de Puente. El nombre del documento es: “Instrucciones_Despliegue.pdf”

Anexo VI - Dockerfiles.

Carpeta donde se encuentran los archivos en formato texto donde se puede ver los distintos caminos para el despliegue de Puente. El nombre del documento es: “dockerfiles”

Anexo VII - Imágenes de presentaciones.

Carpeta donde se encuentran las imágenes de las distintas capacitaciones y presentaciones de la plataforma. El nombre del documento es: “Imágenes”

Anexo VIII - Video de capacitación.

Video en formato “mp4” donde se puede visualizar un tutorial de capacitación sobre la carga de ofertas en la plataforma. El nombre del video es “Video_Capacitacion.mp4”

Anexo IX - Flujo de un match.

Archivo en formato “png” donde se puede visualizar la imagen que contiene el flujo de un match. El nombre del archivo es “FlujoMatch.png”

10 - Bibliografía

1. Modelado BPMN [en línea]. Universidad de Chile. Consultado el 15 de enero de 2024 https://users.dcc.uchile.cl/~nbaloian/DSS-DCC/Software/ModeladoBPMN.pdf

2. Web oficial de Trampoline[en línea]. Trampoline. Consultado el 18 de enero de 2024 https://www.trampoline.network/

3. Guía de Diagrama de Gantt con ejemplos [en línea]. ProjectManager. Consultado el 19 de enero de 2024 https://www.projectmanager.com/guides/gantt-chart

4. Web oficial Notion [en línea]. Notion Labs Inc. Consultado el 8 de mayo de 2023 en https://www.notion.so/es-es

5. Web oficial Git [en línea]. Git. Consultado el 8 de mayo de 2023 en

https://git-scm.com/

6. Web oficial GitHub [en línea]. GitHub. Consultado el 8 de mayo de 2023 en https://github.com/

7. Visión general Cliente-Servidor [en línea]. Mozilla Developer. Consultado el 12 de mayo de 2023 en https://developer.mozilla.org/es/docs/Learn/Server-side/First_steps/Client-Server_overview

8. Estilo de arquitectura de microservicios vs monolítica [en línea]. Amazon. Consultado el 12 de mayo de 2023 en https://aws.amazon.com/es/compare/the-difference-between-monolithic-and-microservices-architecture/#:~:text=Una%20aplicaci%C3%B3n%20monol%C3%ADtica%20se%20ejecuta,del%20entorno%20en%20la%20nube.

9. Web oficial Python [en línea]. Python. Consultado el 8 de mayo de 2023 en https://www.python.org/

10. SPA documentación [en línea]. Microsoft Learn. Consultado el 12 de mayo de 2023 en https://learn.microsoft.com/en-us/entra/identity-platform/index-spa

11. HTML: Lenguaje de etiquetas de hipertexto [en línea]. Mozilla Developer. Consultado el 8 de mayo de 2023 en https://developer.mozilla.org/es/docs/Web/HTML

12. Javascript [en línea]. Mozilla Developer. Consultado el 8 de mayo de 2023 en https://developer.mozilla.org/en-US/docs/Web/JavaScript

13. Server-Side Rendering (SSR) [en línea]. NextJS. Consultado el 8 de mayo de 2023 en https://nextjs.org/docs/pages/building-your-application/rendering/server-side-rendering

14. ¿Qué es una interfaz de programación de aplicaciones (API) [en línea]. AWS. Consultado el 8 de mayo de 2023 en https://aws.amazon.com/es/what-is/api/

15. CSS: Cascading Style Sheets [en línea]. Mozilla Developer. Consultado el 8 de mayo de 2023 en https://developer.mozilla.org/en-US/docs/Web/CSS

16. NodeJS Documentation [en línea]. NodeJS. Consultado el 8 de mayo de 2023 en https://nodejs.org/docs/latest/api/

17. Web oficial NPM[en línea]. NPM. Consultado el 8 de mayo de 2023 en

https://www.npmjs.com/

18. Web oficial Express [en línea]. Express. Consultado el 10 de mayo de 2023 en

https://expressjs.com/es/

19. Web oficial MySQL [en línea]. MySQL. Consultado el 10 de mayo de 2023 en

https://dev.mysql.com/doc/

20. MySQL2 [en línea]. NPM. Consultado el 10 de mayo de 2023 en

https://www.npmjs.com/package/mysql2

21. Nodemailer [en línea]. NPM. Consultado el 10 de mayo de 2023 en

https://www.npmjs.com/package/nodemailer

22. SMTP Documentation [en línea]. SMTP. Consultado el 12 de julio de 2023 en

https://www.smtp.com/resources/api-documentation/

23. ¿Qué es HTTPS? [en línea]. Cloudflare. Consultado el 15 de mayo de 2023 en

https://www.cloudflare.com/es-es/learning/ssl/what-is-https/

24. Web oficial Swagger [en línea]. Swagger. Consultado el 12 de julio de 2023 en https://swagger.io/docs/

25. Multer [en línea]. NPM. Consultado el 10 de mayo de 2023 en

https://www.npmjs.com/package/multer

26. Web oficial Angular [en línea]. ESlint. Consultado el 8 de mayo de 2023 en https://angular.io/

27. Web oficial TypeScript [en línea]. TypeScript. Consultado el 8 de mayo de 2023 en https://www.typescriptlang.org/docs/

28. Web oficial de Bootstrap [en línea]. Bootstrap. Consultado el 15 de junio de 2023 en https://getbootstrap.com/docs/4.1/getting-started/introduction/

29. Web oficial de MaterialUI [en línea]. MaterialUI. Consultado el 12 de junio de 2023 en

https://mui.com/material-ui/

30. Web oficial de Tailwindcss [en línea]. Tailwindcss. Consultado el 15 de junio de 2023 en https://tailwindcss.com/

31. ¿Qué es la Inteligencia Artificial o IA? [en línea]. Google Cloud. Consultado el 2 de octubre de 2023 en https://cloud.google.com/learn/what-is-artificial-intelligence?hl=es-419

32. Web oficial de FastApi [en línea]. FastApi. Consultado el 20 de octubre de 2023 en

https://fastapi.tiangolo.com/

33. SentenceTransformers Documentation [en línea]. SBert. Consultado el 3 de octubre de 2023 en https://sbert.net/

34. Logging [en línea]. Python3 Docs. Consultado el 12 de julio de 2023 en https://docs.python.org/3/library/logging.html

35. ¿Qué es el TF-IDF y cómo mejorar la relevancia de tus textos? [en línea]. Seon. Consultado el 3 de octubre de 2023 en https://www.publisuites.com/blog/tf-idf/

36. ¿Qué es Natural Language Processing (NLP)? [en línea]. Techtarget. Consultado el 2 de octubre de 2023 en https://www.techtarget.com/searchenterpriseai/definition/natural-language-processing-NLP#:~:text=Natural%20language%20processing%20(NLP)%20is,in%20the%20field%20of%20linguistics.

37. ¿Qué son las reglas Heurísticas? [en línea]. Seon. Consultado el 3 de octubre de 2023 de 2023 en https://seon.io/es/recursos/glosario/reglas-heuristicas/#:~:text=Las%20reglas%20heur%C3%ADsticas%20son%20atajos,usan%20para%20realizar%20suposiciones%20fundamentadas.

38. ¿Qué es un modelo Transformers? [en línea]. NVidia. Consultado el 3 de octubre de 2023 en https://la.blogs.nvidia.com/blog/que-es-un-modelo-transformer/

39. ¿Qué son los word embeddings? [en línea]. International American Development Bank Blog. Consultado el 3 de octubre de 2023 en https://blogs.iadb.org/conocimiento-abierto/es/que-son-los-word-embeddings/

40. Web oficial HuggingFace [en línea]. HuggingFace. Consultado el 3 de octubre de 2023 de 2023 en https://huggingface.co/

41. Documentación de "*All Mpnet Base V2*" [en línea]. HuggingFace. Consultado el 3 de octubre de 2023 de 2023 en https://huggingface.co/sentence-transformers/all-mpnet-base-v2

42. Fine Tunning Sentence Transformers 3 [en línea]. AurelIO. Consultado el 20 de octubre de 2023 en https://www.aurelio.ai/learn/sentence-transformers-fine-tuning

43. Oracle Database [en línea]. Oracle. Consultado el 10 de mayo de 2023 en

https://www.oracle.com/database/

44. Ventajas y desventajas de MariaDB [en línea]. Dongee. Consultado el 10 de mayo de 2023 en https://www.dongee.com/tutoriales/ventajas-y-desventajas-de-mariadb/

45. ¿Cómo hacer un diagrama entidad relación? [en línea]. Miro. Consultado el 10 de mayo de 2023 en https://miro.com/es/diagrama/como-hacer-diagrama-entidad-relacion/

46. Web oficial JWT [en línea]. JWT. Consultado el 10 de mayo de 2023 en

https://jwt.io/

47. Bcryptjs [en línea]. NPM. Consultado el 10 de mayo de 2023 en

https://www.npmjs.com/package/brcyptjs

48. Auth Guards in Angular [en línea]. Medium. Consultado el 15 de junio de 2023 en https://medium.com/@jaydeepvpatil225/auth-guards-in-angular-6960950b3c6c

49. Web oficial Postman [en línea]. Postman. Consultado el 8 de mayo de 2023 en https://www.postman.com/

50. mysqldump [en línea]. LinuxTotal. Consultado el 10 de noviembre de 2023 en https://www.linuxtotal.com.mx/index.php?cont=info_admon_021

51. Cron y crontab, explicados [en línea]. DesdeLinux. Consultado el 10 de noviembre de 2023 en https://blog.desdelinux.net/cron-crontab-explicados/

52. ¿Cómo configurar Virtual Hosts en apache?[en línea]. Blog Ahierro. Consultado el 10 de noviembre de 2023 en https://blog.ahierro.es/como-configurar-virtual-hosts-en-apache-y-ubuntu/

53. Mantener los procesos vivos después de cierre de sesión SSH [en línea]. EnMiMaquinaFunciona. Consultado el 10 de noviembre de 2023 en

https://www.enmimaquinafunciona.com/pregunta/62555/mantener-los-procesos-vivos-despues-de-cierre-de-sesion-ssh

54. ¿Qué es SSH? [en línea]. Hostinger. Consultado el 12 de julio de 2023 en https://www.hostinger.com.ar/tutoriales/que-es-ssh

55. Web oficial Let 's Encrypt [en línea]. Let 's Encrypt. Consultado el 12 de julio de 2023 en https://letsencrypt.org/

56. ¿Qué es DNS? [en línea]. CloudFlare. Consultado el 10 de noviembre de 2023 en https://www.cloudflare.com/es-es/learning/dns/what-is-dns/

57. Web oficial Nuthosting [en línea]. Nuthosting. Consultado el 10 de noviembre de 2023 en https://nuthost.com/

58. ¿Qué es cPanel? [en línea]. Hostinger. Consultado el 10 de noviembre de 2023 en https://www.hostinger.com.ar/tutoriales/que-es-cpanel

59. FTP: qué es y cómo funciona [en línea]. Xataka. Consultado el 10 de noviembre de 2023 en https://www.xataka.com/basics/ftp-que-como-funciona

60. Web oficial DonWeb [en línea]. DonWeb. Consultado el 10 de noviembre de 2023 en https://donweb.com/es-ar/

61. Web oficial CentOS [en línea]. CentOS. Consultado el 10 de noviembre de 2023 en https://www.centos.org/download/

62. Guía de Proxy Inverso [en línea]. Apache. Consultado el 10 de noviembre de 2023 en https://httpd.apache.org/docs/trunk/es/howto/reverse_proxy.html

63. Comienzo en Docker [en línea]. Docker. Consultado el 15 de febrero de 2024 en https://docs.docker.com/get-started/overview/

64. Documentación de Ubuntu [en línea]. Ubuntu. Consultado el 10 de noviembre de 2023 en https://help.ubuntu.com/

65. Web oficial de Apache[en línea]. Apache. Consultado el 10 de noviembre de 2023 en https://httpd.apache.org/docs/2.4/es/

66. Web oficial Certbot [en línea]. Certbot. Consultado el 12 de julio de 2023 en https://eff-certbot.readthedocs.io/en/stable/install.html

67. Web oficial Page Speed Insights[en línea]. Docker. Consultado el 15 de febrero de 2024 en https://pagespeed.web.dev/

68. Patrón Backends for Frontends [en línea]. Microsoft. Consultado el 12 de mayo de 2023 en https://learn.microsoft.com/es-es/azure/architecture/patterns/backends-for-frontends

69. Generando saltos y hashes en contraseñas con BcryptJS [en línea]. Medium. Consultado el 15 de mayo de 2023 en https://medium.com/@arunchaitanya/salting-and-hashing-passwords-with-bcrypt-js-a-comprehensive-guide-f5e31de3c40c

70. Boden, M. A. (2018). Artificial Intelligence: A Very Short Introduction. Oxford University Press.

71. 7 maneras de implementar similitud de textos en Python [en línea]. Spotintelligence. Consultado el 2 de octubre de 2023 en https://spotintelligence.com/2022/12/19/text-similarity-python/

72. Word2Vec: Análisis de textos [en línea]. Konfuzio. Consultado el tres de octubre de 2023 en https://konfuzio.com/es/wordtovec/#:~:text=%C2%BFQu%C3%A9%20es%20Word2vec%3F,en%20una%20forma%20matem%C3%A1ticamente%20detectable.

73. Fúquene Ardila, H. J.. (2024). Procesamiento de Lenguaje Natural, los Transformers y los Bots Conversacionales: algunas generalidades. Recuperado de https://www.researchgate.net/publication/382022084_Procesamiento_de_Lenguaje_Natural_los_Transformers_y_los_Bots_Conversacionales. Consultado el 5 de febrero de 2024

74. Deploy aplicación de NodeJS [en línea]. Medium. Consultado el primero de noviembre de 2023 en https://medium.com/@ottoabarriosp/deploy-de-aplicaci%C3%B3n-node-js-4fe1cd68297c

75. ¿Qué es cPanel? [en línea]. Hostinger. Consultado el diez de noviembre de 2023 en https://www.hostinger.com.ar/tutoriales/que-es-cpanel

76. Procedimiento para la elaboración de un análisis FODA [en línea]. Universidad Veracruzana. Consultado el 18 de Enero de 2024 https://www.uv.mx/iiesca/files/2012/12/herramienta2009-2.pdf

77. Lanzan una plataforma para conectar universidades[en línea]. Infobae. Consultado el 18 de Enero de 2024 https://www.infobae.com/educacion/2023/05/30/lanzan-una-plataforma-para-conectar-universidades-de-america-latina-con-empresas-de-todo-el-mundo/

78. Metodologías de Gestión de Proyectos: Modelo en cascada [en línea]. Digital Talent. Consultado el 23 de enero de 2024 https://www.dtagency.tech/cursos/metodologias_gestion_proyectos/tema_1-ModeloWaterfall.pdf

79. Instalar docker en Debian [en línea]. Docker. Consultado el 15 de febrero de 2024 en https://docs.docker.com/engine/install/debian/