



# Práctica Profesional Supervisada

Comunidad de trabajo UNAHUR

Alumnos:      Acuña, Martín  
                    Brandariz, Sebastian  
                    Cuevas, Agustín  
                    Yturrieta, Joel

Docentes:      Lombardi, Carlos  
                    Schiffino, Cristian

2do cuatrimestre 2022

# ÍNDICE

<b>Antecedentes y descripción del proyecto</b>	<b>4</b>
<b>Alcance del proyecto</b>	<b>4</b>
<b>Requerimientos funcionales</b>	<b>5</b>
Roles del sistema:	5
Requerimientos Funcionales comunes a todos los roles:	5
RF1: Registrar Usuario	5
RF2: Iniciar Sesión	6
RF3: Editar Datos de Usuario (Perfil)	6
RF4: Cerrar sesión	7
RF5: Buscar Ofertas	7
RF6: Visualizar Perfil de usuario	7
Requerimientos Funcionales Usuario Postulante	8
RF Postulante 1: Aplicar a Oferta	8
RF Postulante 2: Desaplicar oferta	8
Requerimientos Funcionales Usuario Empresa	9
RF Empresa 1: Crear Oferta	9
RF Empresa 2: Editar Oferta	10
RF Empresa 3: Pausar Oferta	10
RF Empresa 4: Listar Postulantes	11
RF Empresa 5: Visualizar Perfil Postulante	12
Requerimientos Funcionales del usuario Administrador	12
RF Admin 1: Aprobar Empresa	12
RF Admin 2: Aprobar Oferta	13
RF Admin 3: Suspender Oferta	14
<b>Requerimientos no funcionales</b>	<b>14</b>
Usabilidad	14
Seguridad	15
Compatibilidad	15
<b>Metodología de trabajo implementada</b>	<b>15</b>
Avance de los sprint	15
Planilla de inicio de sprint	16
Trello	17
Planilla de fin de sprint	18
Ficha de fin de sprint	19
Diagrama de casos de uso	21
Prototipo Figma	22
Pantalla de inicio de sesión	22

Sesión iniciada	23
Perfil de Usuario Postulante	24
Visualizar una oferta	25
Pop up de postulación	26
Perfil de Administrador	27
Admin - Panel de Ofertas Pendientes	28
<b>Aspectos de la implementación que resultaron desafiantes</b>	<b>29</b>
Almacenamiento de archivos	29
Cifrado de contraseña utilizando bcrypt	29
Barra de búsqueda vs. menú de filtros	30
<b>Diagrama de arquitectura</b>	<b>30</b>
<b>Tecnologías utilizadas</b>	<b>31</b>
Lenguaje	31
Frontend	31
Backend	32
Base de datos	33
Seguridad	33
<b>Relaciones entre pantallas y llamadas a servicios</b>	<b>34</b>
<b>Diagramas DER</b>	<b>34</b>
Diagrama DER de la versión anterior de la base de datos	35
Diagrama DER completo de la versión actual de la base de datos	36
Tablas principales de la versión actual de la base de datos	36
Usuarios:	36
Empresas:	37
Postulantes:	38
Ofertas:	39
<b>Set de pruebas</b>	<b>40</b>
<b>Funcionalidades deseables a implementar en un futuro</b>	<b>47</b>
<b>Conclusiones</b>	<b>47</b>
<b>Bibliografía</b>	<b>47</b>

## Antecedentes y descripción del proyecto

Durante el inicio del primer cuatrimestre del año, en marzo de 2022, en la materia antecesora Desarrollo de Aplicaciones, dentro de las propuestas para desarrollar, nos proponen la creación integral de una aplicación web dedicada a establecer relaciones entre empresas que ofertan empleos y la comunidad interna y externa de la Universidad (como usuarios que buscan empleo). De esta forma, se busca lograr el contacto entre ambas partes para abrir las puertas al ambiente laboral tanto para los estudiantes como para toda la comunidad.

No existía, a la fecha de inicio del proyecto, ningún antecedente de esta aplicación implementada dentro del ámbito de la Universidad Nacional de Hurlingham, lo cual consideramos por demás interesante para iniciar el proyecto desde cero y así poder desarrollar de manera integral una aplicación de tal magnitud.

## Alcance del proyecto

Con esta aplicación buscamos entablar relaciones entre empresas, que ofertan empleos, e individuos, que se encuentren en la búsqueda de estos; a su vez, generar relaciones interpersonales entre estos actores y la UNAHUR.

Por otra parte, permitir que el sistema genere informes para obtener estadísticas de cuántos alumnos, o no alumnos, de la UNAHUR han aplicado a esos empleos, cuantas ofertas propone determinada empresa, cuántos individuos han aplicado a una oferta, entre otras funciones.

# Requerimientos funcionales

## Roles del sistema:

- Postulante
- Empresa
- Administrador

Cada uno de ellos tiene distintos requerimientos aunque algunos son compartidos. A continuación se detalla cuáles son y a quiénes pertenecen.

## Requerimientos Funcionales comunes a todos los roles:

### RF1: Registrar Usuario

Objetivo: El sistema permite el registro de un usuario con perfil postulante o empresa

Entrada: Dirección de correo postulante / empresa (con formato de correo válido), contraseña (mínimo 6 caracteres alfanuméricos, 20 máximo)

Proceso:

- Una vez ingresado el correo y la contraseña, el usuario debe pulsar el botón “SIGUIENTE”, una vez pulsado, el sistema va a redirigir a la siguiente pantalla para continuar con el registro.
- El sistema verifica la veracidad de los datos ingresados y que estos no se encuentren previamente cargados en el registro de usuarios.
- Si el correo ya se encuentra registrado, se va a informar al usuario mediante un mensaje de error.
- Si el correo no se encuentra registrado, el sistema va a redirigir al usuario al siguiente campo para seleccionar el tipo de cuenta.
- Si es un usuario con perfil postulante, deberá seleccionar la opción “POSTULANTE”, el sistema va a dirigir al usuario a la siguiente pantalla donde tendrá que ingresar los datos personales solicitados.
- Si es un usuario con perfil empresa, deberá seleccionar la opción “EMPRESA”, el sistema va a dirigir al usuario empresa a la siguiente pantalla donde tendrá que ingresar los datos comerciales solicitados.

Salida: Usuario Postulante registrado / Usuario Empresa registrado

---

## RF2: Iniciar Sesión

Objetivo: El sistema permite que el usuario ingrese al sistema

Entrada: Dirección de correo postulante / empresa (con formato de correo válido), contraseña (mínimo 6 caracteres alfanuméricos, 20 máximo)

Proceso:

- Una vez ingresado el correo y la contraseña, el usuario debe pulsar el botón “INGRESAR” para que el sistema pueda redirigir al usuario a su perfil.
- El sistema verifica la veracidad de los datos ingresados en el registro de usuarios.
- Si el correo o la contraseña no son correctos, se va a informar al usuario mediante un mensaje de error y no podrá ingresar al sistema.
- Si el correo y la contraseña son correctos, el usuario va a ingresar con éxito a la aplicación.

Salida: Sesión iniciada.

---

## RF3: Editar Datos de Usuario (Perfil)

Objetivo: El sistema permite que un usuario autenticado pueda acceder a su perfil y que tenga la posibilidad de editar sus datos.

Entrada: ID Usuario - Menú de usuario - selección “Perfil” - Datos a modificar (Nombre, Apellido, Fecha de Nacimiento, Tipo de Documento, Número de Documento, Nacionalidad, Provincia, Localidad, Dirección, Altura de calle, Teléfono de Contacto).

Proceso:

- El sistema permite que el usuario autenticado una vez dentro de la pantalla de perfil, tenga la posibilidad de editar sus datos.
- Una vez editados los datos que desee cambiar, en la parte inferior tendrá la posibilidad de seleccionar entre dos botones “Guardar” que guarda los cambios realizados, y “Cancelar”, que descarta los cambios realizados persistiendo los datos que ya se encontraban registrados.

Salida: Confirmación de grabación OK

---

#### RF4: Cerrar sesión

Objetivo: El sistema permite que el usuario logueado pueda cerrar la sesión de su cuenta

Entrada: ID Usuario

Proceso:

- El sistema permite que el usuario autenticado pueda cerrar sesión al seleccionar el botón del logo, una vez seleccionado, se despliega el menú con las opciones de usuario, donde encontrará la opción “Cerrar Sesión”.
- El usuario debe presionar el botón “Cerrar Sesión”, el sistema mostrará un pop up con la leyenda “¿Está seguro que desea cerrar su sesión?” junto a las opciones de Cancelar o Confirmar.
- Una vez confirmado, el sistema redirigirá al usuario a la pantalla inicial de la aplicación.

Salida: Sesión cerrada

---

#### RF5: Buscar Ofertas

Objetivo: El sistema permite que el usuario logueado pueda buscar una oferta con alguna palabra específica.

Entrada: ID Usuario , ingresar palabra a buscar en el buscador.

Proceso:

- El sistema permite que el usuario pueda filtrar las ofertas con el buscador, una vez ingresada la palabra o parte de la palabra a buscar debe presionar el icono “lupa” en la barra de búsqueda.

Salida: Listado de ofertas filtradas.

---

#### RF6: Visualizar Perfil de usuario

Objetivo: El sistema permite que el usuario logueado pueda visualizar su perfil.

Entrada: ID Usuario

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará menú de usuario.
- En el menú desplegado, deberá seleccionar “Perfil”.

Salida: Visualización de perfil de usuario.

---

## Requerimientos Funcionales Usuario Postulante

### RF Postulante 1: Aplicar a Oferta

Objetivo: Permitir que el usuario pueda postularse a una oferta.

Entrada: ID Usuario autenticado - ID Oferta

Proceso:

- El usuario deberá seleccionar una oferta de su interés, dentro de las ofertas visualizadas mediante aplicado o no de filtro de búsqueda. (Descrito en RF anterior).
- Dentro de la oferta aparecerá el botón “Postularme”, el cual el usuario deberá seleccionar.
- El sistema mostrará un pop up con la leyenda “Deseas postularte a “Título de la oferta”” y los botones “Sí, postularme” y “No, cancelar”.
- Una vez seleccionado el botón “Sí, postularme”, el pop up cambiará la leyenda a “Te has postulado a “Título de la oferta”, Buena suerte”, y un botón la leyenda “OK”, que el usuario deberá seleccionar para cerrar el pop up.

Salida: Usuario postulado a la oferta.

---

### RF Postulante 2: Desaplicar oferta

Objetivo: El usuario postulado a una oferta pueda desaplicar de la misma.

Entrada: ID Usuario autenticado - ID Oferta

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.



- En el menú desplegado, el usuario deberá seleccionar la opción “Postulaciones”.
- En el listado obtenido de las ofertas a las que aplicó, deberá seleccionar la oferta de la cual desea desaplicar.
- Una vez dentro de la oferta aplicada, el sistema le dará la opción al usuario de desaplicar a la oferta.
- Una vez seleccionada la oferta a desaplicar, el sistema mostrará un pop up con la leyenda “Desea eliminar su postulación a “Título de la oferta”” y los botones “Eliminar” y “Cancelar”.
- El usuario deberá seleccionar la opción “Eliminar”.
- El sistema desplegará otro pop up confirmando la eliminación de la postulación con la leyenda “Eliminaste tu postulación a “Título de la oferta””.

Salida: Usuario desaplicado de la oferta.

---

## Requerimientos Funcionales Usuario Empresa

### RF Empresa 1: Crear Oferta

Objetivo: El sistema permite que una Empresa pueda crear una oferta laboral.

Entrada: ID Usuario Empresa autenticado - ID Empresa - Datos de oferta (Nombre de la oferta, descripción, fecha de vigencia, horario laboral desde/hasta, edad desde/hasta, experiencia previa descripción, zona de trabajo, áreas de estudio, otros detalles, beneficios, remuneración, estudio, carrera, jornada, contrato)

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, el usuario deberá seleccionar “Perfil empresa”.
- En la nueva pantalla el usuario observará 3 opciones, “Datos”, “Ver Ofertas” y “Crear Ofertas”, opción esta última que deberá seleccionar.
- Una vez seleccionada, el sistema desplegará una pantalla con los campos que el usuario debe completar para crear una oferta.
- Una vez completados todos los campos, el usuario deberá seleccionar el botón “Confirmar”. Si algún campo de los obligatorios no es completado, el botón “Confirmar” permanecerá grisado.

Salida: Oferta creada pendiente de publicación.

---

## RF Empresa 2: Editar Oferta

Objetivo: El sistema permite que el usuario pueda modificar una oferta ya creada.

Entrada: ID Usuario Empresa autenticado - ID Empresa - Datos de oferta (Nombre de la oferta, descripción, fecha de vigencia, horario laboral desde/hasta, edad desde/hasta, experiencia previa descripción, zona de trabajo, áreas de estudio, otros detalles, beneficios, remuneración, estudio, carrera, jornada, contrato)

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, deberá seleccionar “Perfil empresa”.
- En la nueva pantalla, el usuario observará 3 opciones, “Datos”, “Ver Ofertas” y “Crear Ofertas”, deberá seleccionar “Ver Ofertas”.
- Una vez dentro de la pantalla desplegada, el usuario observará un listado con todas las ofertas creadas, deberá seleccionar la oferta que desee modificar seleccionando la opción “Ver Oferta”.
- Dentro de la pantalla desplegada, tendrá la opción de seleccionar el botón “Editar Oferta”.
- Una vez seleccionada la opción, ingresará a la pantalla con todos los datos de la oferta donde podrá modificar los campos que el usuario desee.
- Se deberán completar los campos obligatorios a modificar, si un campo obligatorio no se completó correctamente aparecerá una ventana emergente avisando esto.
- Una vez terminada la modificación, el usuario selecciona el botón “Confirmar”.

Salida: Oferta actualizada.

---

## RF Empresa 3: Pausar Oferta

Objetivo: El sistema permite que el usuario logueado pueda suspender una oferta ya creada.

Entrada: ID Usuario Empresa - ID Oferta

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, el usuario deberá seleccionar “Perfil empresa”.
- En la nueva pantalla observará 3 opciones, “Datos”, “Ver Ofertas” y “Crear Ofertas”, deberá seleccionar “Ver Ofertas”.
- Una vez dentro de la pantalla desplegada, el usuario observará un listado con todas las ofertas creadas, deberá seleccionar en la oferta que desee suspender el botón “Pausar”.
- Una vez seleccionada la opción, aparecerá una pestaña con la opción de confirmar la acción o cancelar.
- Selecciona el botón “Confirmar”.

Salida: Oferta pausada.

---

## RF Empresa 4: Listar Postulantes

Objetivo: El sistema permite que el usuario pueda obtener el listado de los postulantes que aplicaron a una oferta ya creada.

Entrada: ID Empresa - ID Oferta

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, el usuario deberá seleccionar “Perfil empresa”.
- En la nueva pantalla observará 3 opciones, “Datos”, “Ver Ofertas” y “Crear Ofertas”, el usuario deberá seleccionar “Ver Ofertas”.
- Una vez dentro de la pantalla desplegada, el usuario observará un listado con todas las ofertas creadas, deberá seleccionar la oferta de la cual desee ver el listado de postulados a esa oferta.
- El sistema lo redirigirá a la pantalla de la oferta donde deberá seleccionar la opción “Ver postulantes”.

Salida: Listado de postulantes con sus datos de contacto.

---

## RF Empresa 5: Visualizar Perfil Postulante

Objetivo: El sistema permite que el usuario pueda visualizar el perfil de los postulantes a una oferta.

Entrada: ID Usuario Empresa - ID Postulante

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, el usuario deberá seleccionar “Perfil empresa”.
- En la nueva pantalla observará 3 opciones, “Datos”, “Ver Ofertas” y “Crear Ofertas”, deberá seleccionar “Ver Ofertas”.
- Una vez dentro de la pantalla desplegada, el usuario observará un listado con todas las ofertas creadas, deberá seleccionar la oferta de la cual desee ver el listado de postulados a esa oferta, seleccionando la opción “Ver postulantes”.
- Dentro de la pantalla desplegada, el usuario observará los datos de Nombre, DNI y Teléfono de los postulantes, también tendrá la opción de seleccionar el botón “Ver”.
- Habiendo seleccionado la opción “Ver”, ingresará a los datos del perfil del postulante donde podrá observar los datos personales.

Salida: Listado de postulantes con sus datos de contacto y acceso a datos personales.

---

## Requerimientos Funcionales del usuario Administrador

### RF Admin 1: Aprobar Empresa

Objetivo: Aprobar una Empresa para que pueda operar en el sistema.

Entrada: Usuario Administrador autenticado

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, el usuario deberá seleccionar “Panel Administrador”.

- En la nueva pantalla el usuario observará 4 opciones, “Postulantes”, “Ofertas”, “Empresas” y “Estadísticas”. Deberá seleccionar “Empresas”
- Una vez seleccionada la opción, el sistema desplegará una pantalla con el listado de Empresas y los datos básicos de las mismas (CUIT, Nombre de Empresa, Representante y mail de contacto) y observará un botón “Aceptar Empresas” que deberá seleccionar.
- En la pantalla que se despliega, el usuario observará un listado de empresas pendientes de aprobación con sus correspondientes datos: CUIT, Nombre de Empresa, Representante y mail de contacto y observará un botón “Aceptar” que deberá seleccionar.
- El sistema desplegará un pop-up que solicitará confirmación para activar la empresa en el sistema.
- Aparecerá un pop-up que confirmará que la empresa fué activada.

Salida: Empresa activada para operar en el sistema.

---

## RF Admin 2: Aprobar Oferta

Objetivo: Aprobar una oferta creada por una Empresa.

Entrada: Usuario Administrador autenticado - ID Oferta

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, el usuario deberá seleccionar “Panel Administrador”.
- En la nueva pantalla observará 4 opciones, “Postulantes”, “Ofertas”, “Empresas”, y “Estadísticas”. El usuario deberá seleccionar la opción “Ofertas”.
- Una vez seleccionada, el sistema desplegará una pantalla con el listado de Ofertas y los datos básicos de las mismas (ID Oferta, Título, Empresa, Estado), el usuario observará un botón “Aceptar Ofertas” que deberá seleccionar.
- En la pantalla que se despliega, observará un listado de Ofertas pendientes de aprobación con sus correspondientes datos: ID Oferta, Título, Empresa, Estado y el usuario observará un botón “Aceptar Ofertas” que deberá seleccionar

- El sistema desplegará en una ventana nueva, el listado de ofertas pendientes de aprobación junto con dos botones “Ver Oferta” y “Aceptar”, este último es el que deberá seleccionar.
- El sistema desplegará un pop-up que pedirá confirmación del usuario para activar la oferta en el sistema.
- El sistema desplegará un pop-up que confirmará que la oferta fué activada.

Salida: Oferta aprobada.

---

### RF Admin 3: Suspende Oferta

Objetivo: Suspende una oferta creada por una Empresa.

Entrada: ID Usuario Administrador - ID Oferta

Proceso:

- El usuario deberá seleccionar el logo en el margen derecho de la pantalla, el cual desplegará un menú.
- En el menú desplegado, el usuario deberá seleccionar la opción “Panel Administrador”.
- En la nueva pantalla observará 4 opciones, “Postulantes”, “Ofertas”, “Empresas”, y “Estadísticas”. El usuario deberá seleccionar la opción “Ofertas”.
- Una vez seleccionada, el sistema desplegará una pantalla con el listado de Ofertas y los datos básicos de las mismas (ID Oferta, Título, Empresa, Estado) y el usuario observará un botón “Suspende” que deberá seleccionar.
- El sistema desplegará un pop-up para confirmar que la oferta será desactivada.
- Una vez aceptado por el usuario, la oferta se desactivará.

Salida: Oferta suspendida.

## Requerimientos no funcionales

### Usabilidad

Se apuntó a lograr una buena experiencia de usuario al utilizar una interfaz amena, facilitando el primer contacto al permitir visualizar las publicaciones sin necesidad de

autenticación. La aplicación implementa botones e íconos con diseños limpios e intuitivos que facilitan la navegación por las diferentes páginas.

## Seguridad

Desde el inicio del proyecto, definimos el uso de librerías creadas para facilitar la utilización de hashing y tokens, de esta manera se permite la securización del sistema, librerías tales como Bcrypt que realiza el hashing de la contraseña antes de persistir en la base de datos, sumado al empleo de un *salt* para aumentar la aleatoriedad del hash o el uso de un token generado por la librería JWT que nos permite la sustitución de un elemento sensible como la contraseña por otro equivalente y volátil.

## Compatibilidad

Nos enfocamos en utilizar tecnologías probadas y con un grado de madurez alto para asegurar la compatibilidad con los navegadores más utilizados en la actualidad. Para ello utilizamos el compilador Babel que nos permite desarrollar usando los estándares de ECMAScript pero manteniendo la compatibilidad con motores más viejos.

## Metodología de trabajo implementada

Decidimos utilizar metodología Agile, compuesta por Scrum combinada con Kanban con sprints de 3 semanas (iteraciones que fueron de 3 semanas, límite máximo de feedback de producto real y reflexión) y seguimiento semanal por parte tanto de los docentes como del stakeholder. El uso de metodologías Ágiles sumado a las tarjetas de Kanban nos permitió mayor autonomía del grupo, y facilitan la priorización y asignación de las tareas a realizar y así poder mostrar avances de forma semanal.

## Avance de los sprint

Para el desarrollo de cada sprint se contó con las siguientes herramientas: La planificación inicial a través de un documento de inicio de sprint, el seguimiento de las tareas del sprint mediante tarjetas en Trello y un documento de fin de sprint.

## Planilla de inicio de sprint

Los documentos se armaron entre los equipos de estudiantes y docentes. En un principio se guiaba sobre el alcance de cada sprint, con el paso de las semanas, las guías eran cada vez menores y el equipo lograba un mayor autonomía. A continuación se brinda como ejemplo, la primera planilla de inicio de sprint, y la planilla de fin de sprint. El diseño de éstas fue constante en los sprint sucesivos.

### Desarrollo de aplicaciones - 1er cuatrimestre 2022

#### Ficha de principio de sprint

Grupo - proyecto		
Grupo 1 - Comunidad de trabajo		

Nro de sprint	Fecha de comienzo	Fecha de fin
1	07-04-2022	28-04-2022

Objetivos que nos ponemos para este sprint (una frase para cada uno).

- Hacer una 1ra versión de maquetado en Figma (Desktop).
- Hacer una 1ra versión de maquetado en Figma (Mobile).
- Definir requerimientos para la aplicación con Carlos (Stakeholder).
- Definir el stack de tecnologías a utilizar con Carlos (profe).
- Analizar qué modificaciones deberíamos realizar sobre la base.
- Averiguar cómo funciona la API de consulta de alumnos.
- Definir filtros de las ofertas con Carlos (StakeHolder).
- Conseguir el MER/DER de la base.
- Crear organización en Github.



## Trello

Es una herramienta en línea que permite gestionar proyectos a través de tableros y tarjetas, la estructura utilizada por nosotros fue:

Hecho en este sprint: Para archivar las tarjetas de los objetivo finalizados

Estamos haciendo ahora: Aquí se ponen las tarjetas de las actividades en las que el equipo trabaja actualmente.

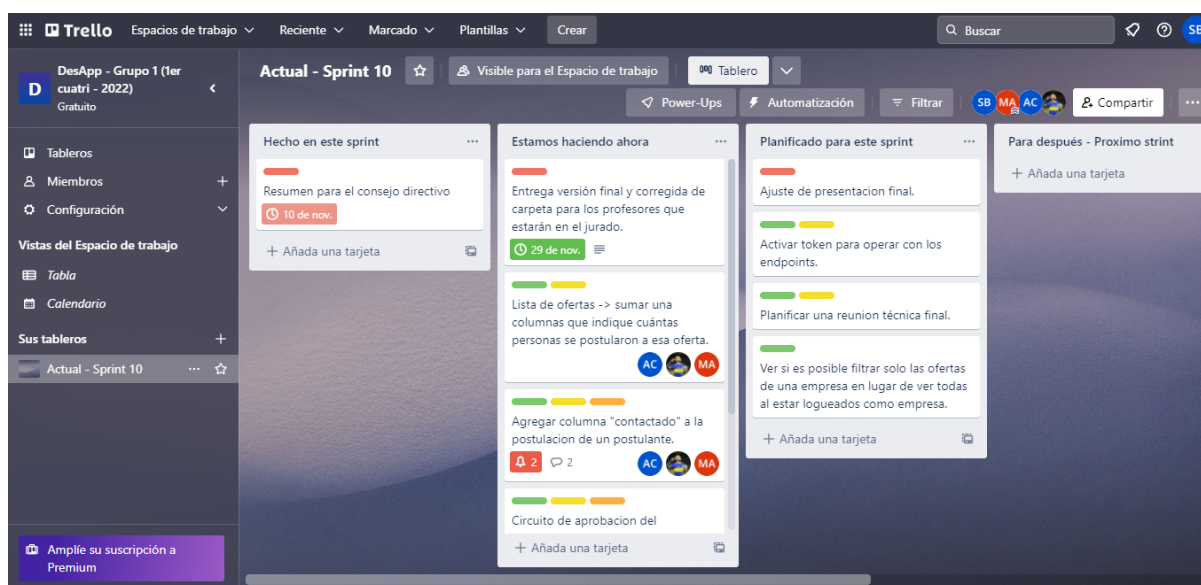
Planificado para este sprint: Aquí se listan todas las tareas planificadas para el sprint actual.

Para después - Próximo sprint: Aquí se colocan las actividades que surgen y son necesarias de implementar, pero que no están contempladas en el sprint actual, luego pasan a formar parte del siguiente.

Para más adelante: Son tareas que surgen y quedan para una próxima planificación.

En resumen, las tarjetas son desplazadas de un lugar a otro, cualquier persona del equipo puede acceder al tablero y, de una simple mirada, ver en qué estado se encuentran las tareas actuales, qué tarea se terminó y qué tarea queda por hacer.

Ejemplo de Trello: Las imágenes muestran un tablero a modo de ejemplo de lo descrito anteriormente.



Al finalizar un sprint, el tablero se cierra y queda archivado en “Pasado”, en el mismo, se guardan las listas de lo hecho en cada sprint. Así se ven los primeros cuatro tableros correspondientes a los 4 primeros sprints.



## Planilla de fin de sprint

Para el cierre del sprint, se presenta un documento con los objetivos alcanzados con una frase concreta y un porcentaje, y de lo que no se pudo finalizar, junto a contar brevemente cuál o cuáles fueron los motivos para no llegar a cumplir las metas. Se deja a continuación, y a modo de ejemplo, el documento de fin de sprint del primer sprint.

## Ficha de fin de sprint

### Desarrollo de aplicaciones - 1er cuatrimestre 2022

#### Ficha de fin de sprint

Grupo - proyecto
Grupo 1 - Comunidad de trabajo

Nro de sprint	Fecha de comienzo	Fecha de fin
1	07-04-2022	28-04-2022

Objetivos que nos pusimos para este sprint (una frase para cada uno).

- Hacer una 1ra versión de maquetado en Figma (Desktop).
- Hacer una 1ra versión de maquetado en Figma (Mobile).
- Definir requerimientos para la aplicación con Carlos (Stakeholder).
- Definir el stack de tecnologías a utilizar con Carlos (profe).
- Analizar qué modificaciones deberíamos realizar sobre la base.
- Averiguar cómo funciona la API de consulta de alumnos.
- Definir filtros de las ofertas con Carlos (StakeHolder).
- Conseguir el MER/DER de la base.
- Crear organización en Github.

A qué llegamos, o nos acercamos bastante.

**Hacer una 1ra versión de maquetado en Figma (Desktop/Mobile) - 100%.**

Le presentamos a Carlos (StakeHolder) la primera versión del maquetado y quedó conforme con esta versión. Durante la charla definimos los pasos a seguir para la segunda versión del maquetado.

**Definir requerimientos para la aplicación con Carlos (Stakeholder) - 100%.**

Se definieron requerimientos para las funcionalidades del usuario y administrador.

**Definir el stack de tecnologías a utilizar con Carlos (profe) - 100%.**

Vamos a utilizar las tecnologías recomendadas para la materia:

React JS (Material UI, Bootstrap) / Node JS (Sequelize) / PostGresql

También definimos que vamos a usar, dentro de la organización de GITHUB dos repos, uno para front y otro para back. Ya que creemos que esto facilita la organización del grupo.

**Analizar qué modificaciones deberíamos realizar sobre la base - 100%.**

Al no disponer del código actual de la aplicación decidimos descartar el diseño de la base y crear todas las entidades y relaciones según nuestro criterio. Pensando que en un futuro vamos a reemplazar las funcionalidades existentes por nuestro desarrollo.

El DER inicial se encuentra compartido por Google Drive.

**Averiguar cómo funciona la API de consulta de alumnos - 100%.**

En la reunión que tuvimos con Carlos como profe nos compartió la estructura de consulta de la API del SIU para analizarlo y ver cuales consultas vamos a necesitar.

*Carlos no comenta que actualmente están en vías de crear una maqueta con datos ofuscados para realizar consultas de prueba.*

**Definir filtros de las ofertas con Carlos (StakeHolder) - 100%.**

Durante la charla con Carlos como StakeHolder definimos los filtros para las ofertas, estos son los que se nos ocurrieron en la charla pero no es necesario limitarse solo a estos.

Filtros iniciales:

- Zona urbana (Ubicación)
- Tipo de trabajo Full-time/Part-time
- Competencias (si está incluida)
- Requiere licencia de conducir
- Requiere experiencia previa
- Requiere conocimientos previos (ej: office, Linux, windows, etc.)
- Secundario completo
- Cursos / Idiomas / Intereses

**Conseguir el MER/DER de la base - 100%.**

Nos lo compartió Cristian durante la clase.

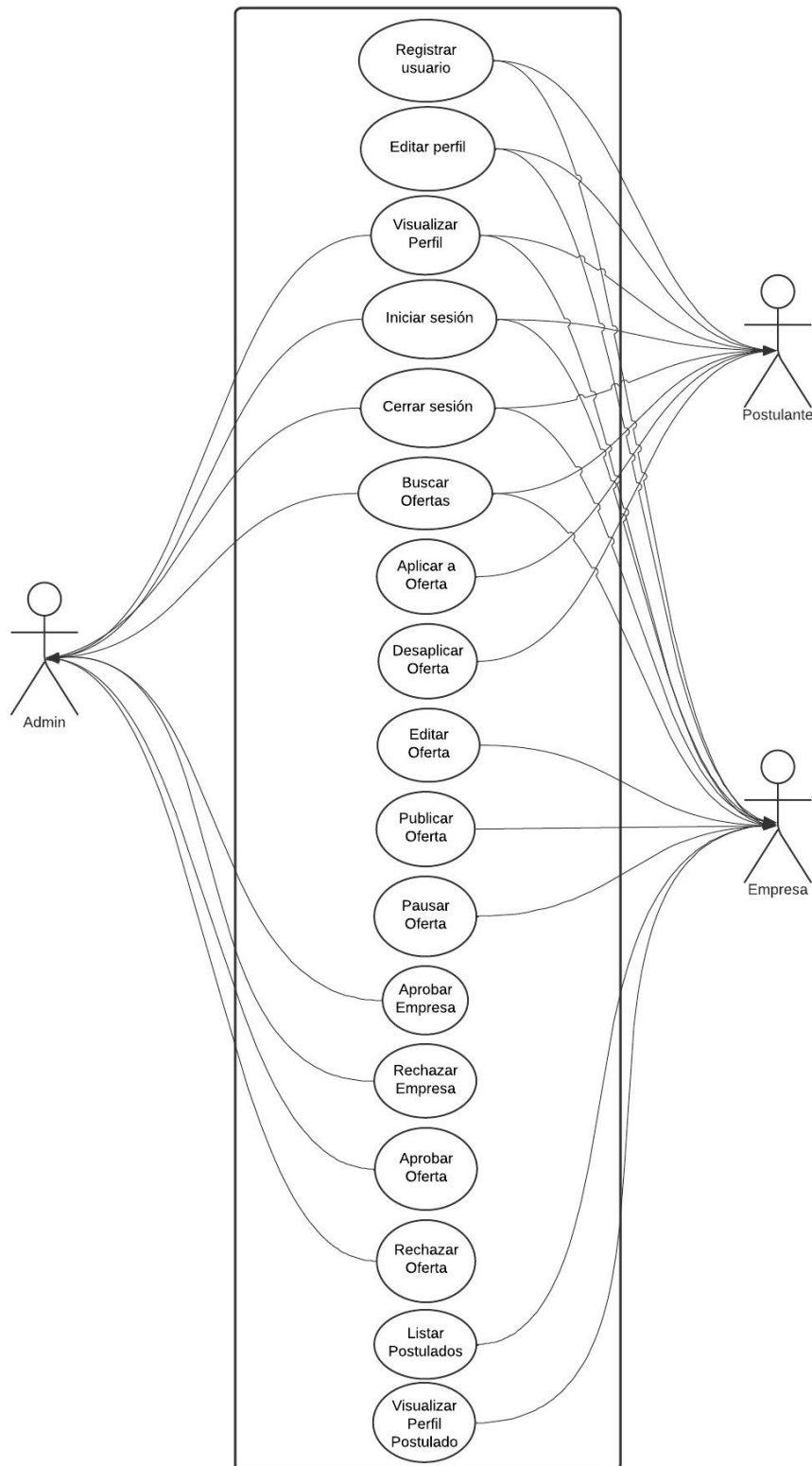
**Crear organización en Github - 100%.**

Creamos la organización con los accesos y estructura acorde.

**A qué no llegamos:**

Para cada cosa a la que no se llegó, contar brevemente **por qué**, o sea, qué problemas aparecieron.

## Diagrama de casos de uso



Para información más detallada se ofrecen las plantillas utilizadas en cada sprint y la información de los tableros.:

[Plantillas de inicio y fin de sprint](#). Las plantillas de Inicio se realizaban y generaban en la planning, las planillas de finalización con las tareas que se lograron finalizar y el detalle de aquellas que por distintos motivos, pasaban a ser parte de la planificación siguiente.

[Tablero Sprint Pasado](#): Contiene la información de lo hecho en cada sprint en la plataforma Trello.

## Prototipo Figma

Para el inicio del proyecto se realizó un prototipo en Figma con la funcionalidad del sistema propuesto. A continuación se observan las pantallas del prototipo y finales para realizar una comparación, y así, ver la evolución que realizó el equipo al desarrollar las mismas.

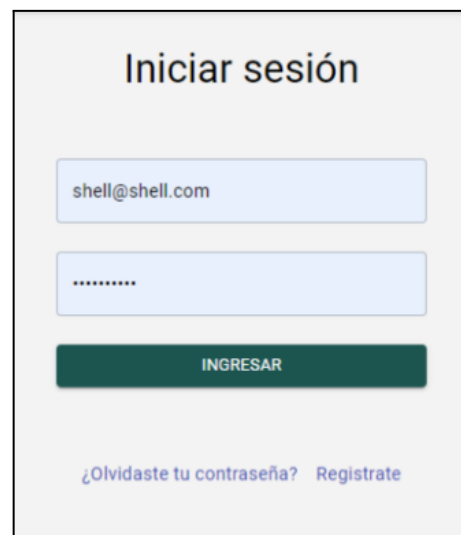
### Pantalla de inicio de sesión

Versión Figma:



The Figma version of the login screen is a wireframe. It features a title 'Inciar Sesion' at the top. Below it are two input fields: the first is preceded by a green user icon and labeled 'Usuario'; the second is preceded by a green lock icon and labeled 'Contraseña'. Below these fields is a checkbox labeled 'Recordar Contraseña'. A blue button labeled 'Ingresar' is positioned below the checkbox. At the bottom, there is a link that says '¿Todavía no te registraste?'.

Versión real:



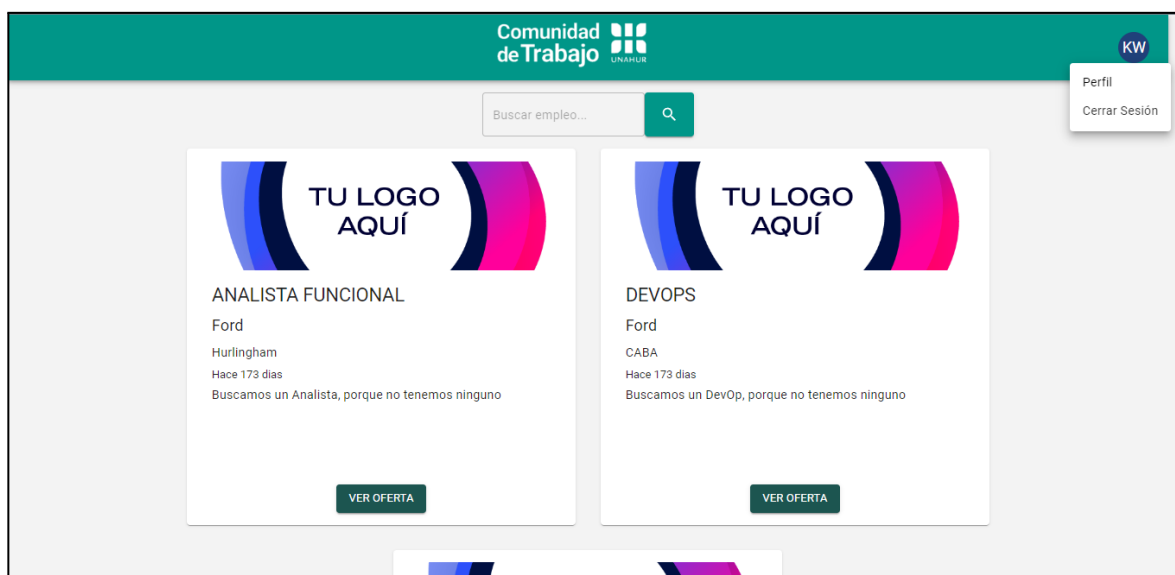
The real version of the login screen is a polished design. It has a title 'Iniciar sesión' at the top. Below it are two light blue input fields; the first contains the email 'shell@shell.com' and the second contains masked characters '\*\*\*\*\*'. Below these fields is a dark green button labeled 'INGRESAR'. At the bottom, there is a link that says '¿Olvidaste tu contraseña? Registrate'.

Sesión iniciada

Versión Figma:



Versión real:

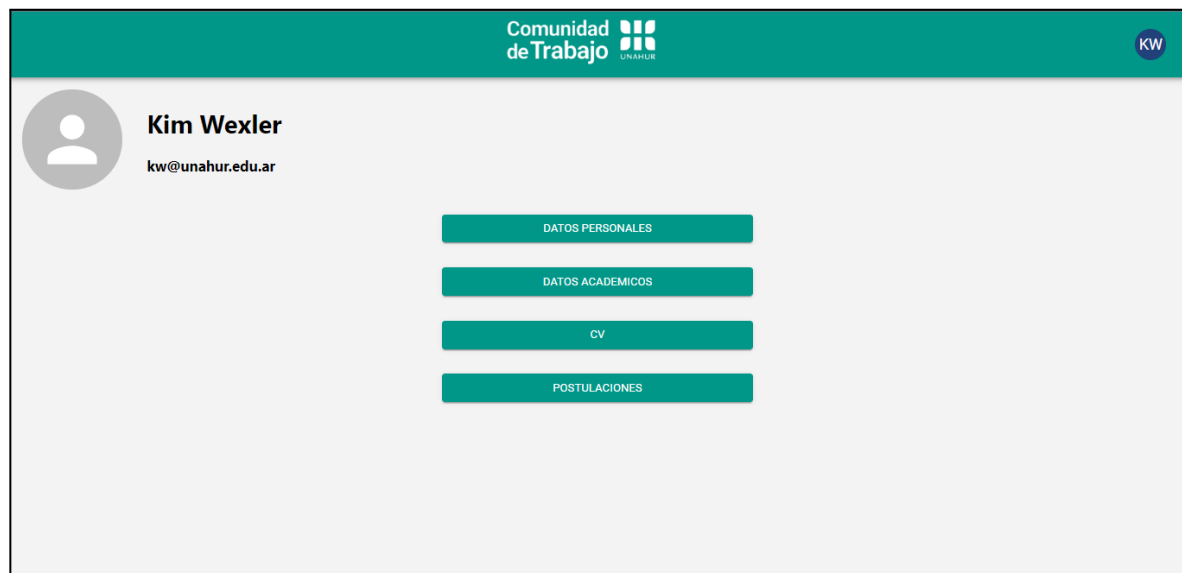


## Perfil de Usuario Postulante

Versión Figma:



Versión real:





## Visualizar una oferta

Versión Figma:



Versión real:



## Pop up de postulación

Versión Figma:



Versión real:

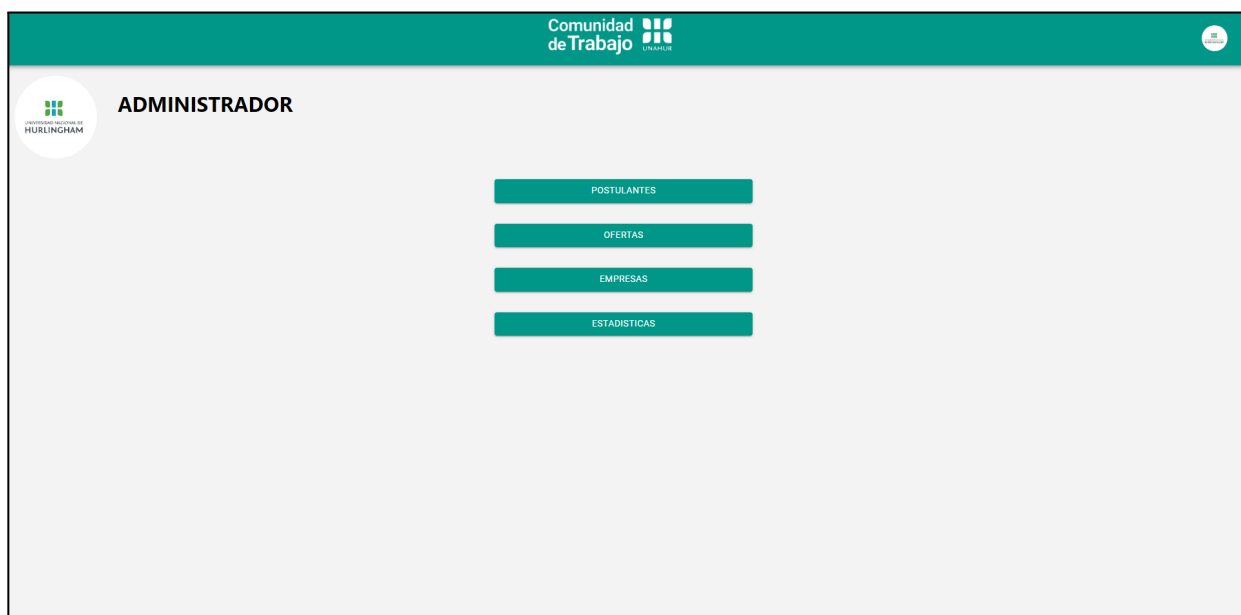


## Perfil de Administrador

Versión Figma:



Versión real:



## Admin - Panel de Ofertas Pendientes

Versión Figma:




Ofertas pendientes

Filtrar por

Jornada

▼

Edad desde

Desde

Edad hasta

Hasta

Contrato

▼

Experiencia

☐

Idioma

▼

Ubicacion

▼

Fecha publicacion

▼

Buscar empleo

🔍



**Desarrollador Java JR**  
 Nombre empresa  

Aceptar Rechazar



**Desarrollador Java JR**  
 Nombre empresa  

Aceptar Rechazar



**Desarrollador Java JR**  
 Nombre empresa  

Aceptar Rechazar



**Desarrollador Java JR**  
 Nombre empresa



**Desarrollador Java JR**  
 Nombre empresa



**Desarrollador Java JR**  
 Nombre empresa

Versión real:

Comunidad de Trabajo 				
<div> <div>Buscar ofertas...</div> <div>🔍</div> </div>				
ID Oferta	Titulo	Empresa	Estado	Acciones
52	Gerente de Marketing	Shell	pausada	<div>VER OFERTA</div> <div>ACEPTAR</div> <div>SUSPENDER</div>
<div>&lt; 1 &gt;</div>				

# Aspectos de la implementación que resultaron desafiantes

## Almacenamiento de archivos

El manejo de archivos y el resguardo de los mismos no fue una tarea sencilla de resolver. Luego de haber definido la estructura de la base de datos planteamos la idea de convertir los archivos a base64, que lo transforma de binario a un formato ASCII y así poder insertarlo en la base. Esto conlleva varios inconvenientes como por ejemplo la base de datos es menos eficiente que un file system por lo que generalmente tarda más en grabar y leer datos almacenados. El tamaño de los archivos hace que la base crezca de forma desmesurada y esto genera que las consultas desde la aplicación se ralenticen provocando un problema de performance.

Por estos motivos decidimos utilizar el servicio de Google Cloud Storage, más adelante en este documento describiremos el servicio, pero como ventaja podemos decir que, al ser un servicio en la nube, facilita el resguardo de los archivos y nos libera de realizar backups de la información, por otro lado el servicio ofrece medidas de seguridad como control de usuario y certificados para autenticación, además de almacenamiento gratis hasta 5Gb por mes de transferencia.

Una vez definido el medio que íbamos a utilizar avanzamos con la creación de los endpoints para la gestión de los archivos, revisamos la documentación publicada por Google para implementar el servicio, realizamos los ajustes en la base para almacenar el registro de los archivos y seguimos avanzando hasta resolver el problema. Ya creado el endpoint continuamos con el código del frontend, para terminar con las pruebas necesarias.

## Cifrado de contraseña utilizando bCrypt

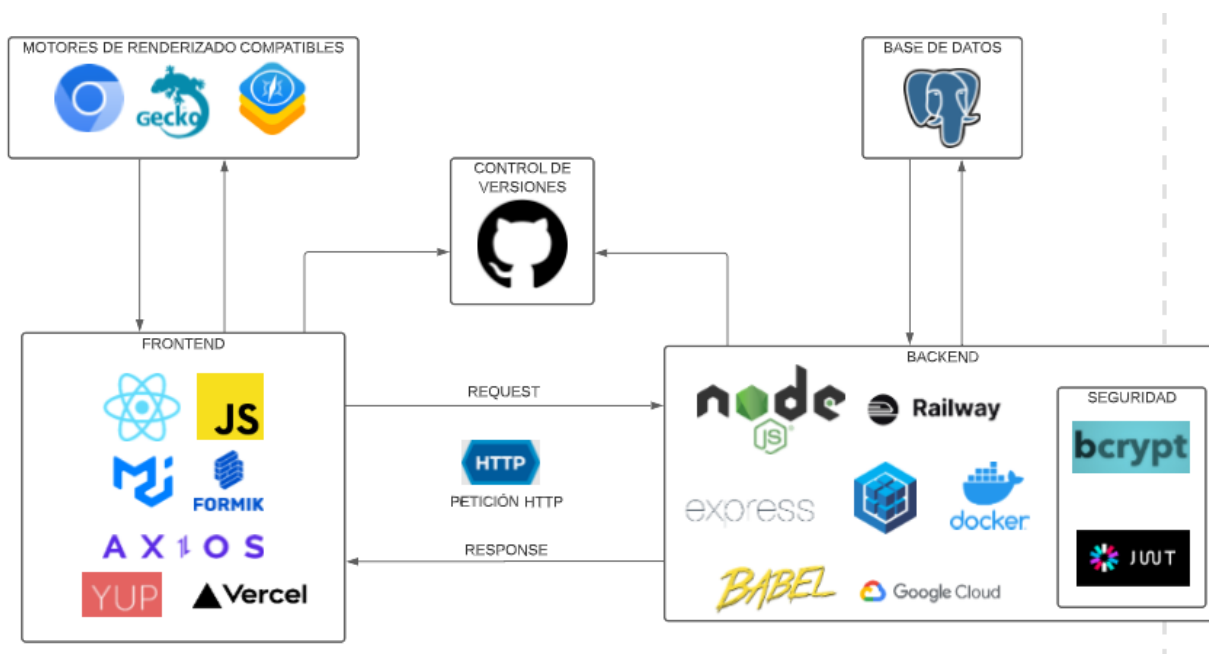
Desde un primer momento nos enfocamos en fortalecer la seguridad del desarrollo, uno de los componentes utilizados y que requirió de un análisis previo a la implementación, fue el uso de la librería bCrypt. Esta librería permite el realizar hashing de la contraseña antes de persistir en la base de datos, sumado al empleo de un salt que utiliza una palabra clave para aumentar la aleatoriedad del hash creado y evita que una misma contraseña creada por diferentes usuarios obtengan un mismo hash. Realizamos el

análisis de la librería leyendo la documentación disponible e implementamos un endpoint de testing, en ese punto nos dimos cuenta que era posible guardar directamente el hash en lugar de la contraseña así que luego de hacer las modificaciones necesarias en la tabla de usuario desplegamos el servicio en el entorno de producción.

## Barra de búsqueda vs. menú de filtros

La implementación de la barra de búsqueda, en lugar de la búsqueda por filtros que se propuso en un principio fue otro de los puntos que requirió el análisis de todo el equipo, debido a que no solo generaría la modificación en el código sino un cambio estético que sería visto por el usuario. Inclinarlos por la barra de búsqueda nos permitió mostrar la página principal mucho más limpia y presentar la información al usuario de forma directa. En cuanto al desarrollo, realizamos la búsqueda en el backend permitiendo que desde el frontend, mediante la utilización de *query params*, filtren, orden o limiten las búsquedas sin que el usuario final tenga que aplicar filtros de forma manual. Como beneficio extra podemos decir que al aplicar los filtros desde el backend solo se enviará la información requerida por el usuario minimizando los datos transferidos a este.

## Diagrama de arquitectura



# Tecnologías utilizadas

## Lenguaje

**JavaScript:** Es un poderoso lenguaje de programación que permite incorporar dinamismo a los sitios web. Se utiliza tanto del lado del servidor, en nuestro caso al utilizar el entorno NodeJS, como del lado del cliente en el navegador. La utilización del mismo lenguaje en ambos lados nos brinda uniformidad y facilita la revisión del código por cualquier miembro del equipo.

## Control de versión

**GitHub:** Es un servicio que permite alojar proyectos utilizando el sistema de control de versiones Git. La mayor ventaja de esta herramienta es permitir el trabajo colaborativo.

**Git:** Es un sistema de control de versiones distribuido. Está pensado en la eficacia y la confiabilidad del mantenimiento de versiones de aplicaciones optimizado para guardar cambios incrementales. Lleva control del historial y permite llevar registro de los cambios realizados por otros desarrolladores.

## Frontend

**React:** Es una de las librerías más utilizadas en la actualidad, escrita en JavaScript y desarrollada por Facebook para facilitar la creación de componentes interactivos, reutilizables, para interfaces de usuario. Permite que las vistas de las pantallas se asocien con los datos, de modo que si cambian los datos, también cambian las vistas. De las funcionalidades disponibles en la versión 18.1.0 se destacan los componentes funcionales y Hooks, que nos permiten desarrollar aplicaciones con una mejor experiencia de usuario.

**Material UI:** Es una librería de componentes más utilizadas junto a React que nos ayuda a diseñar una interfaz de usuario de forma mas rapida y sencilla, con un diseño moderno y con tecnología Responsive que permite observar la aplicación de manera correcta en una PC, tablet, celular o cualquier otro tipo de dispositivo.

**Vercel:** Es una plataforma en la nube para sitios estáticos y serverless que permite alojar sitios y servicios web. Esta plataforma nos permitió trabajar de manera dinámica

deployando el frontend y de esta forma permitir que todo el equipo acceda a nuevas funcionalidades de la aplicación sin importar desde dónde se conectan y poder observar el funcionamiento como lo haría el usuario final.

**Formik:** Esta librería, se utiliza para la creación e implementación de formularios en el frontend, para validar formularios con JavaScript, que da soporte a aplicaciones de navegador.

**Yup:** Esta librería nos permite validar los valores introducidos en cada campo de los formularios antes de enviarlos al backend.

**Axios:** Es una librería JavaScript que nos permite de forma sencilla realizar solicitudes a una API utilizándolo como cliente HTTP.

## Backend

**NodeJs:** Es un entorno multiplataforma de código abierto para la capa del servidor basado en JavaScript que utiliza eventos asíncronos. Nos permite crear aplicaciones distribuidas de una manera más sencilla.

**ExpressJs:** Es una librería que permite crear APIs y aplicaciones web fácilmente, provee un conjunto de características como manejo de rutas (direccionamiento), archivos estáticos, uso de motor de plantillas, integración con bases de datos, manejo de errores entre otras funcionalidades.

**Sequelize:** Sequelize es un ORM para Nodejs que nos permite convertir tablas de una base de datos, en objetos de un lenguaje de programación orientado a objetos, en nuestro caso JavaScript, lo cual agiliza bastante el acceso a estos datos. De esta manera permite manipular varias bases de datos SQL de forma sencilla.

**Babel:** Es un transpilador gratis y de código abierto, que nos permite desarrollar usando los estándares de ECMAScript pero manteniendo la compatibilidad con motores más viejos. Ya que al ejecutarse el servidor el código se transpila transformando la sintaxis a versiones compatibles de JS.

**Google Cloud Storage:** Es un servicio del tipo infraestructura como servicio (IaaS) para almacenamiento de archivos en la nube de tipo RESTful. El servicio combina el rendimiento y la escalabilidad de la nube de Google. Ofrece medidas de seguridad como



control de usuario y certificados para autenticación, además de almacenamiento gratis hasta 5Gb por mes de transferencia. Con este servicio implementamos el endpoint para el manejo de archivos en la aplicación.

**Docker:** Esta herramienta permite automatizar el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción, virtualizando el sistema operativo pero solo usando los recursos necesarios. Con Docker creamos un contenedor de la base de datos para el entorno de pruebas y así poder levantarla sin mucho esfuerzo en cada computadora donde se clonara el repositorio.

**Heroku y Railway:** Estas dos plataformas en la nube ofrecen servicios para desplegar base de datos y aplicaciones directamente desde el repositorio en GitHub, aceptan diferentes lenguajes de programación. En nuestro caso desplegamos el backend y la base de datos PostgreSQL, en un primer momento en Heroku pero el servicio pasó a ser pago a mediados de noviembre, esto nos obligó a buscar otra plataforma para realizar la migración del backend. En la actualidad el servidor y la base están desplegados en Railway.

## Base de datos

**PostgreSQL:** Es un sistema de gestión de bases de datos relacionales (RDBMS) libre y de código abierto, motor de bases de datos compatible con los servicios de cloud y la mayoría de las herramientas más utilizadas. Es compatible con diversos modelos de datos para crear aplicaciones orientadas a objetos. Este gestor de bases de datos posee un control de concurrencias multiversión.

## Seguridad

**bCrypt:** Es una herramienta de encriptado para utilizar en contraseñas. El principal beneficio que conlleva el uso de **bCrypt** es que aumenta la seguridad, ya que realiza el hashing de la contraseña antes de persistir en la base de datos, sumado al empleo de un *salt* para aumentar la aleatoriedad del hash.

**JWT:** Es una herramienta de estándar abierto cuyo objetivo es establecer la transmisión de información entre dos o más participantes de forma segura firmándola para certificar

que cada cual es quien dice ser. Para esto genera un token que nos permite la sustitución de un elemento sensible como la contraseña por otro equivalente y volátil.

## Relaciones entre pantallas y llamadas a servicios

Requerimiento funcional	Pantalla	Endpoint
Registrar usuario	/registroUsuario	/usuarios/signup/
Inciar sesion	/login	/usuarios/signin/
		/postulantes/idUsuario/\${data.id}
		/empresas/idUsuario/\${data.id}
Editar datos de usuario	/miPerfil/misDatos/editar	/postulantes/dni/\${id}
		/empresas/cuit/\${id}
Buscar ofertas	/	/ofertas/buscarTitulo=\${tituloOferta}
Visualizar perfil de usuario	/postulante/\${id}	/postulantes/dni/\${id}
Aplicar a oferta	/oferta/\${id}	/postulaciones/
Desaplicar oferta	/ofertasPostulante	/postulaciones/\${id}
Crear oferta	/RegistroOferta	/ofertas/
Editar oferta	/edicionOferta/\${id}	/ofertas/idOferta/\${id}
Pausar oferta	/listadoOfertasEmpresa	/ofertas/idOferta/\${id}
Listar postulantes	/ListadoDePostulantes/\${id}	/postulacionesId/oferta/?id=\${id}
Visualizar perfil de postulante	/postulante/\${id}	/postulantes/dni/\${id}
Aprobar empresa	/admin/listadoEmpresasInactivas	/empresas/cuit/\${id}
Aprobar oferta	/admin/listadoOfertasInactivas	/ofertas/idOferta/\${id}
Suspender oferta	/admin/listadoOfertas	/ofertas/idOferta/\${id}

Estos son los endpoints principales. El detalle de la API con cada una de las llamadas a los endpoints con sus consultas y respuestas se puede revisar en el documento anexo: *Documentación API Comunidad de trabajo*.

## Diagramas DER

El proyecto ya contaba con una versión preliminar. Luego de realizar un primer análisis determinamos reestructurar la base de datos tratando de respetar las formas de normalización. Con esa premisa rediseñamos la base para lograr una estructura acorde a la escalabilidad del proyecto, llevándola de su forma inicial a una tercera forma normal.

Primera forma normal (1FN):

- La tabla contiene una clave primaria única.
- La clave primaria no contiene atributos nulos.
- No hay filas duplicadas.

- Cada intersección de fila/columna contiene exactamente un valor del dominio aplicable.
- Todas las columnas son regulares, es decir, las filas no tienen componentes como IDs de fila, IDs de objeto, o timestamps ocultos.

Segunda forma normal (2FN):

Para alcanzar la 2FN, además de cumplir la 1FN, todos los atributos secundarios de una tabla deben ser dependientes de una clave primaria.

Tercera forma normal (3FN):

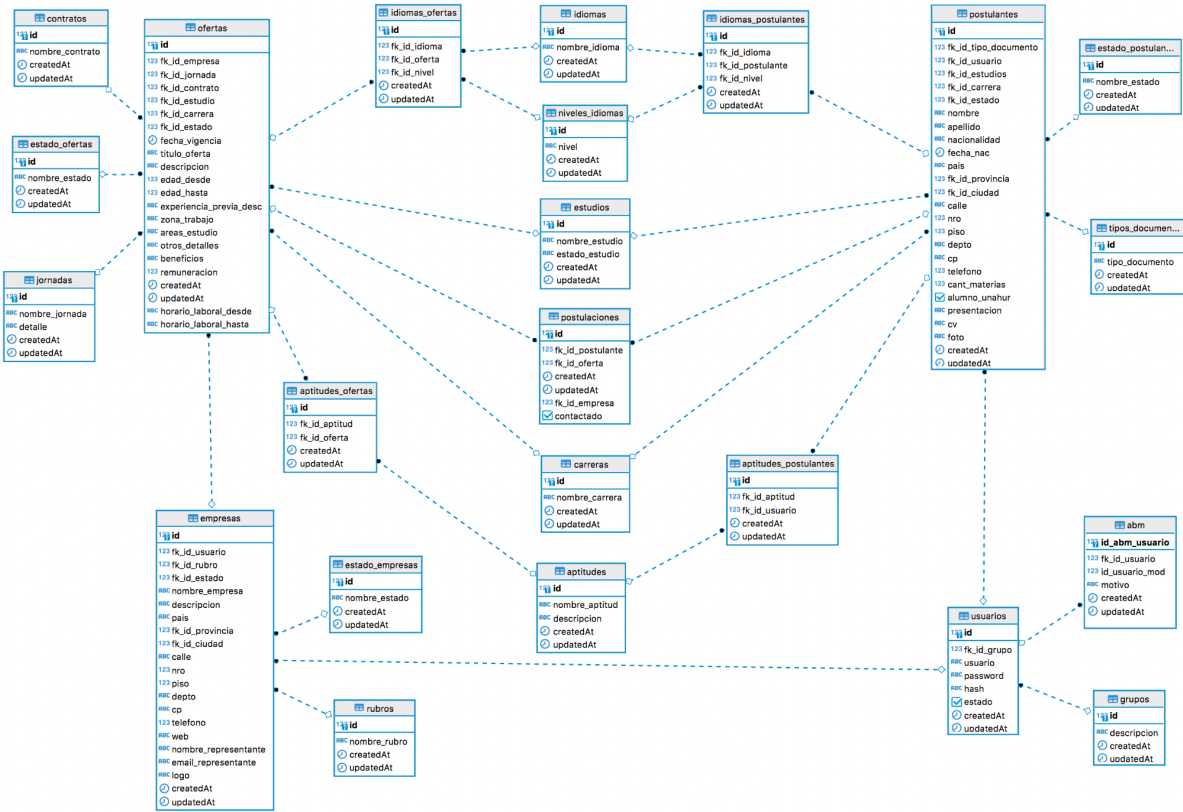
Para alcanzar esta forma diseñamos la base para que ningún atributo secundario de las tablas sea dependiente “transitivamente de” una clave primaria.

Sabiendo que es posible llegar hasta la 5FN normal decidimos que al haber alcanzado la 3FN es suficiente para mantener el orden estructural y obtener una base performante.

## Diagrama DER de la versión anterior de la base de datos

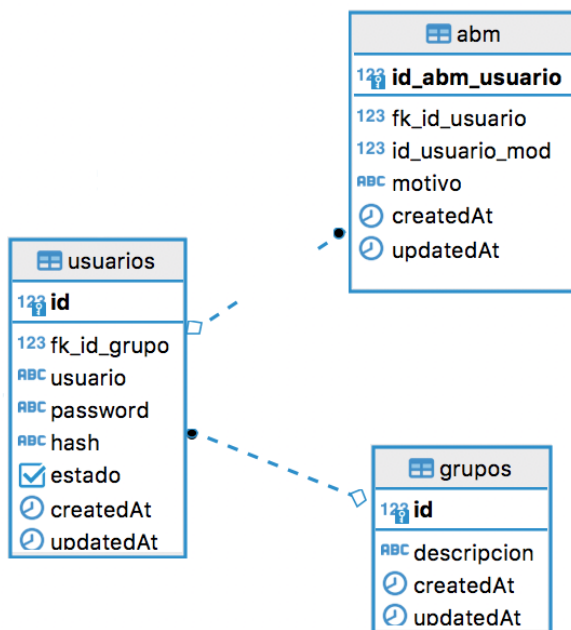


## Diagrama DER completo de la versión actual de la base de datos

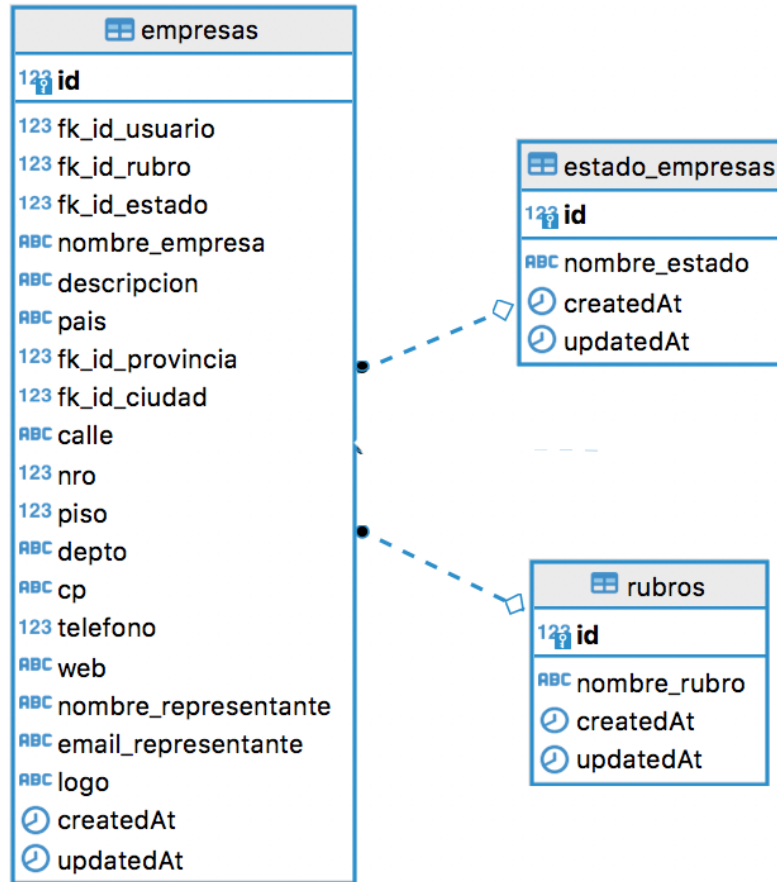


## Tablas principales de la versión actual de la base de datos

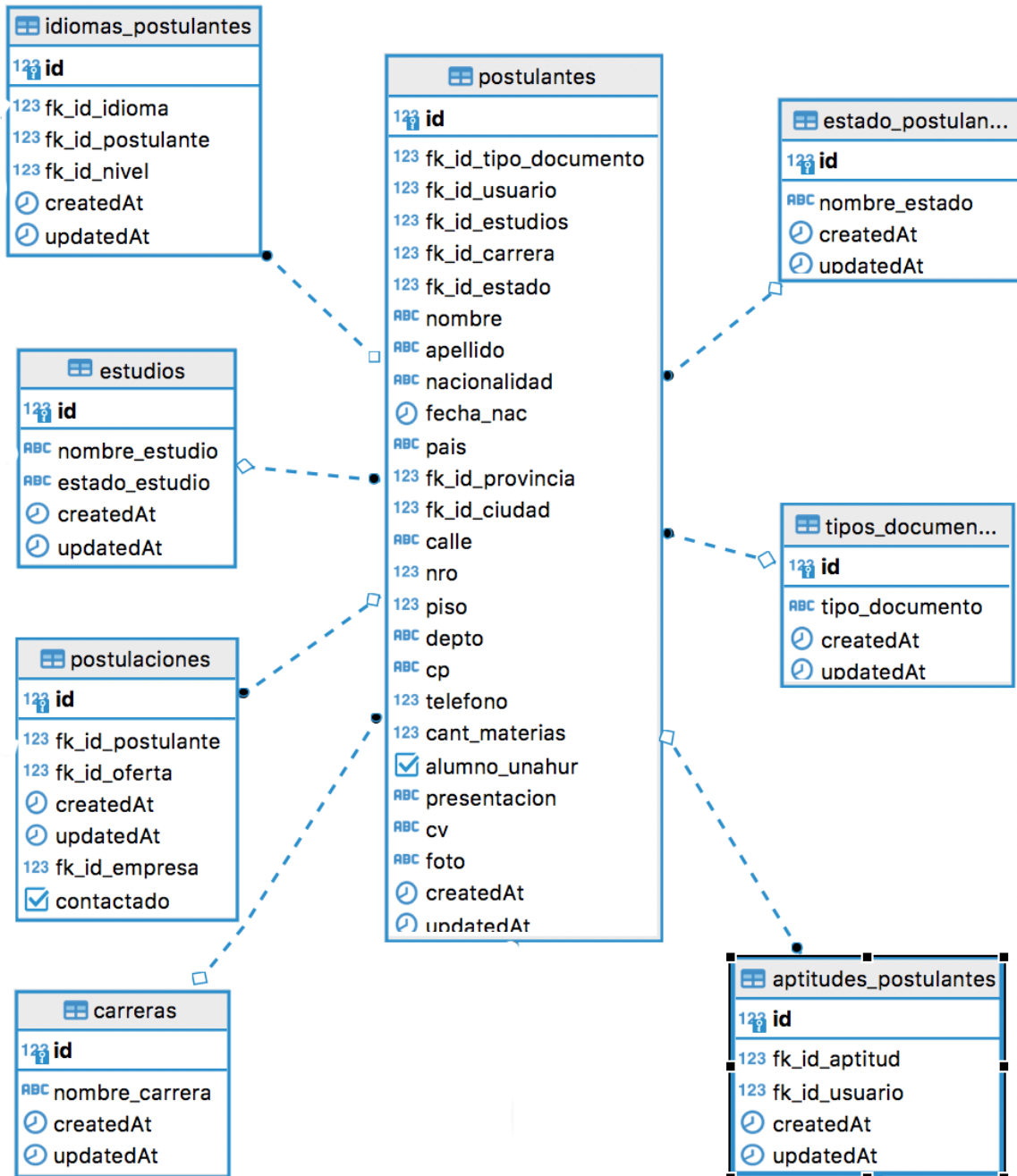
## Usuarios:



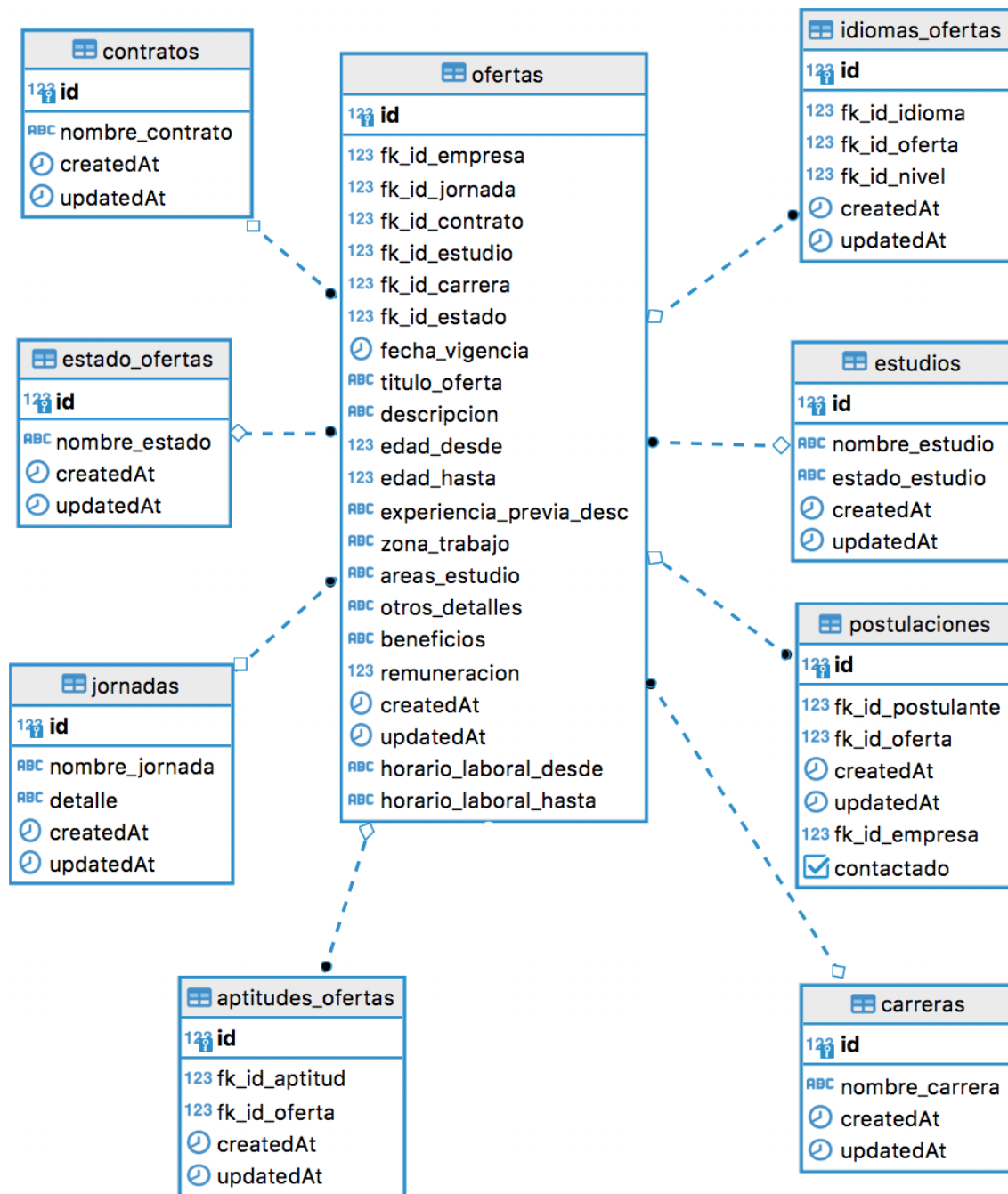
Empresas:



## Postulantes:



## Ofertas:



## Set de pruebas

El objetivo de las pruebas funcionales es validar el comportamiento del sistema, donde se verifica si cumple o no con sus especificaciones. El plan de pruebas se organiza como un conjunto de casos de prueba que derivan de los requisitos funcionales.

Caso de prueba N° 1	Iniciar sesión.
Descripción	Se verifica la entrada al sistema a través de la pantalla de logueo, se valida el mail y la contraseña del usuario.
Prerrequisitos	<ul style="list-style-type: none"> <li>El usuario debe estar registrado en la aplicación.</li> </ul>
Datos de entrada	<ul style="list-style-type: none"> <li>Usuario Email.</li> <li>Contraseña de usuario.</li> </ul>
Datos de salida	Acceso correcto del usuario al sistema.
Resultados de la prueba	Logueo del usuario exitoso.

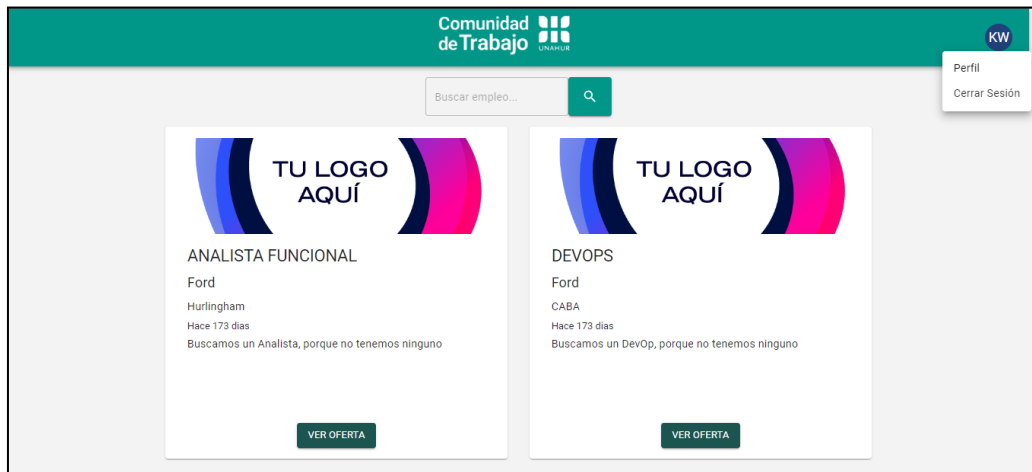
Iniciar sesión:

### Iniciar sesión

[¿Olvidaste tu contraseña?](#)   [Regístrate](#)



Sesión iniciada:



Caso de prueba N° 2	Buscar oferta.
Descripción	El usuario puede buscar una oferta por su nombre.
Prerrequisitos	<ul style="list-style-type: none"> <li>• Debe existir una oferta.</li> </ul>
Datos de entrada	<ul style="list-style-type: none"> <li>• Nombre de la oferta.</li> </ul>
Datos de salida	Listado de ofertas que coinciden con el dato de entrada.
Resultados de la prueba	Listado de ofertas filtradas.




Caso de prueba N° 3	Postulación a una oferta.
Descripción	El usuario postulante puede postularse a una oferta.
Prerrequisitos	<ul style="list-style-type: none"> <li>• Debe existir una oferta.</li> <li>• No debe existir una postulación del usuario en la oferta.</li> <li>• El usuario debe tener la sesión iniciada .</li> </ul>
Datos de entrada	<ul style="list-style-type: none"> <li>• Nombre de la oferta.</li> <li>• Seleccionar el botón “postularme”.</li> <li>• Confirmar postulación en el pop-up.</li> </ul>
Datos de salida	El usuario confirma la postulación a la oferta.
Resultados de la prueba	Usuario postulado.

## Desarrollador Jr4

Robertolandia Modificado 5

📍 Zona oeste

📅 Publicado hace: 95 días



### Sobre la empresa

soy la descripción de una empresa

### Descripción

Tenes que hacer muchas cosas

### Horario

🕒 De 8hs a 18hs

### Contrato

Freelance

### Salario

\$100000

### Beneficios

Muchos beneficios

**POSTULARME**



¿Deseas postularte a  
Desarrollador Jr4?

¡Si, postularme!

No, cancelar



Te has postulado a Desarrollador  
Jr4

¡Buena suerte!

OK

Caso de prueba N° 4	Crear oferta.
Descripción	El usuario empresa puede crear una oferta.
Prerrequisitos	<ul style="list-style-type: none"> <li>La empresa debe estar activa dentro de la aplicación.</li> </ul>
Datos de entrada	<ul style="list-style-type: none"> <li>ID Usuario Empresa autenticado - ID Empresa - Datos de oferta (Nombre de la oferta, descripción, fecha de vigencia, horario laboral desde/hasta, edad desde/hasta, experiencia previa descripción, zona de trabajo, áreas de estudio, otros detalles, beneficios, remuneración, estudio, carrera, jornada, contrato).</li> </ul>
Datos de salida	La oferta se crea con estado pausado.
Resultados de la prueba	Oferta creada.

### Datos de oferta

Nombre de la oferta

Desarrollador JAVA SSR.

Horario laboral desde:

800

Edad hasta:

50

Áreas de estudio

Informatica

Remuneración

380000

Jornada

2 : Full-time

Fecha de vigencia

31/12/2022

Edad desde:

20

Zona de trabajo

CABA


Beneficios

Varios

Carrera

2: Licenciatura en informatica

CONFIRMAR



**La oferta fue creada exitosamente**

Para continuar pulse el boton

Finalizar

Desarrollador JAVA SSR.

Shell

pausada


VER

VER POSTULANTES

BORRAR

Caso de prueba N° 5	Editar datos de perfil.
Descripción	El usuario postulante puede editar sus datos de perfil en la aplicación.
Prerrequisitos	<ul style="list-style-type: none"> <li>El usuario debe estar logueado en el sistema.</li> </ul>
Datos de entrada	<ul style="list-style-type: none"> <li>Provincia.</li> <li>Ciudad.</li> <li>Nombre de Calle.</li> <li>Altura de calle.</li> <li>Teléfono de contacto.</li> </ul>
Datos de salida	Actualización de datos del usuario.
Resultados de la prueba	Datos de perfil de usuario actualizados.

### Datos personales



**Su registro fue realizado correctamente**

Para continuar pulse el boton

Caso de prueba N° 6	Activar una oferta.
Descripción	El usuario administrador puede aprobar una oferta en la aplicación.
Prerrequisitos	<ul style="list-style-type: none"> <li>• Debe existir una oferta.</li> <li>• El usuario administrador debe estar logueado en el sistema.</li> </ul>
Datos de entrada	<ul style="list-style-type: none"> <li>• Panel de administrador -&gt; ofertas -&gt; ofertas pendientes.</li> </ul>
Datos de salida	Aceptar oferta pendiente.
Resultados de la prueba	Oferta publicada en el sistema.

ID Oferta	Título	Empresa	Estado	Acciones
55	Desarrollador JAVA SSR.	Shell	pausada	<a href="#">VER OFERTA</a> <a href="#">ACEPTAR</a> <a href="#">SUSPENDER</a>

ID Oferta	Título	Empresa	Estado	Acciones
55	Desarrollador JAVA SSR.	Shell	activa	<a href="#">VER OFERTA</a> <a href="#">DESHABILITAR</a>

## Funcionalidades deseables a implementar en un futuro

- Conectar la aplicación a la API de SIU Guaraní.
- Implementar el envío de mails para verificación/recuperación de cuentas.
- Implementar estadísticas.

## Conclusiones

Durante el transcurso de esta cursada integramos lo aprendido en la Tecnicatura aplicando conocimientos de materias como Desarrollo de Aplicaciones, Elementos de Ingeniería de Software, Base de datos, Objetos I, II, Laboratorio de S.O./redes, Construcción de Interfaces de Usuario y Persistencia, entre otras. Trabajamos como equipo aplicando metodologías Ágiles, Scrum y Kanban. Algo importante a destacar es que analizamos e implementamos nuevas tecnologías actualmente utilizadas en la industria.

Tenemos la esperanza de que la aplicación sea usada en el ámbito de la universidad y esto ayude a relacionar a las empresas con la universidad y la comunidad. Como resaltaron varias veces nuestros profesores en las diferentes reuniones que mantuvimos: **“El peor código es el que no llega a producción”**.

## Bibliografía

- <https://www.javascript.com/>
- <https://github.com/>
- <https://mui.com/>
- <https://es.reactjs.org/>
- <https://vercel.com/#get-started>
- <https://formik.org/>
- <https://www.npmjs.com/package/yup>
- <https://axios-http.com/>
- <https://nodejs.org/en/>
- <https://expressjs.com/es/>
- <https://sequelize.org/>

- <https://babeljs.io/>
- <https://cloud.google.com/>
- <https://www.docker.com/>
- <https://railway.app/>
- <https://www.postgresql.org/>
- <https://www.npmjs.com/package/bcrypt>
- <https://jwt.io/>
- <https://www.figma.com/>
- <https://lucid.app/>
- <https://www.google.com/intl/es/drive/>
- <https://docs.google.com/>