



**Universidad
Rey Juan Carlos**

**GRADO EN INGENIERÍA EN TECNOLOGÍAS DE
TELECOMUNICACIÓN**

Curso Académico 2020/2021

Trabajo Fin de Grado

**APLICACIÓN HTML5 PARA INTERCAMBIO DE
INFORMACIÓN LABORAL PRIVADA ENTRE
PROFESIONALES TIC**

Autor : Vicente Giménez García

Tutor : Pedro de las Heras Quirós

Trabajo Fin de Grado

Aplicación HTML5 para Intercambio de Información Laboral Privada entre
Profesionales TIC

Autor : Vicente Giménez García

Tutor : Pedro de las Heras Quirós

La defensa del presente Proyecto Fin de Carrera se realizó el día de
de 2021, siendo calificada por el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de 2021

*Dedicado a
mi familia*

Agradecimientos

Quiero dar sinceras gracias a mi familia y amigos, por apoyarme tanto durante estos años. También a mi profesor Pedro de las Heras Quirós por su tiempo e interés, y por su guía, gracias a la cuál fue posible hacer este proyecto.

Resumen

Una de las dificultades a las que se enfrentan los alumnos en período de prácticas empresariales o recién egresados de la universidad es el desconocimiento de qué salario podría considerarse justo para un determinado puesto ante una oferta de trabajo. A esta dificultad se añade el hecho de que muchos trabajadores consideran hablar de su sueldo un tema tabú.

El objetivo de este Trabajo Fin de Grado es desarrollar una aplicación que proporcione a estudiantes y trabajadores del sector de las tecnologías de la información y la comunicación un modo de evaluar cuál es el rango salarial justo para una determinada profesión u oferta de trabajo teniendo en cuenta la empresa donde se realizará la labor, el puesto de trabajo y las tecnologías relacionadas con el puesto mediante el intercambio de experiencias laborales entre usuarios, permitiendo a éstos mantener el control sobre la privacidad de sus datos y de su identidad.

El proyecto consiste en una aplicación web desarrollada con la librería de JavaScript React y un servidor de aplicaciones escrito en Java con Sprint Boot siguiendo el patrón de arquitectura hexagonal. El resultado es una aplicación que puede ejecutarse en un navegador web y que hace uso de los servicios de intercambio de experiencias laborales que proporciona el servidor. El servidor además está integrado con la plataforma GitHub [29] como proveedor de autenticación y de datos de usuarios, por lo que cualquier usuario con cuenta de GitHub puede hacer uso de la plataforma.

El proyecto ha finalizado cumpliendo sus objetivos, proporcionando al usuario una herramienta donde puede subir sus experiencias laborales eligiendo para cada dato de ésta si es visible al resto de usuarios o no, vinculando estas experiencias a su perfil o publicándolas de forma anónima en la herramienta, permitiéndole hacer búsquedas de experiencias laborales de otros usuarios donde podrá ver la información que estos marcaron como pública y podrá solicitar acceder a la información privada a los usuarios dueños de ésta, a cambio de su propia información privada, mediante una negociación.

Índice general

1. Introducción	1
1.1. Definición de experiencia laboral	2
1.2. Privacidad	2
1.3. Solicitud de información a otros usuarios	3
1.4. Estado del arte	3
1.5. Estructura de la memoria	4
2. Objetivos	5
2.1. Objetivo general	5
2.2. Objetivos específicos	5
2.3. Requisitos	6
2.4. Planificación temporal	10
3. Tecnologías y herramientas	13
3.1. Servidor de aplicaciones	13
3.1.1. Lenguaje y plataforma	13
3.1.2. Comunicación con el cliente	14
3.1.3. Base de datos	14
3.2. Aplicación web	15
3.2.1. Lenguaje y librerías	15
3.2.2. Estilo	16
3.3. Autenticación	16
3.4. Herramientas	17
3.4.1. IntelliJ IDEA	17

3.4.2. Visual Studio Code	17
3.4.3. Maven	17
3.4.4. NPM	18
3.4.5. Git	18
3.5. Arquitectura hexagonal	18
4. Diseño e implementación	21
4.1. Guía de lectura	21
4.2. Arquitectura general	22
4.3. Modelo de dominio	24
4.3.1. Clases e interfaces comunes	25
4.3.2. Objetos de valor del dominio de las experiencias laborales	27
4.3.3. Entidad experiencia laboral	27
4.3.4. Objetos de valor del dominio de las negociaciones	29
4.3.5. Entidad negociación	29
4.4. Modelo de datos	30
4.5. Diseño de la API REST	33
4.5.1. Puntos de API de sesión	34
4.5.2. Puntos de API de datos de referencia	34
4.5.3. Puntos de API de experiencias laborales	36
4.5.4. Puntos de API de negociaciones	40
4.6. Componentes web	44
4.6.1. ActionDisplay	44
4.6.2. AppRouter	45
4.6.3. Autocomplete	45
4.6.4. AutocompleteMultiple	46
4.6.5. CardSkeleton	47
4.6.6. Checkbox	47
4.6.7. Chip	48
4.6.8. ClosableChip	48
4.6.9. DateInput	49

4.6.10. Modal	50
4.6.11. MoneyInput	50
4.6.12. NavbarUser	51
4.6.13. NegotiableWorkExperience	52
4.6.14. NegotiationSummary	53
4.6.15. OwnWorkExperience	53
4.6.16. Radio	54
4.6.17. RequestButton	55
4.6.18. Select	55
4.6.19. WorkExperience	56
4.6.20. WorkExperienceSummary	56
4.7. Vistas	57
4.7.1. Inicio de sesión	58
4.7.2. Bienvenida	58
4.7.3. Mis experiencias	58
4.7.4. Añadir experiencia	63
4.7.5. Editar experiencia	63
4.7.6. Buscar experiencias	66
4.7.7. Solicitud de información	66
4.7.8. Mis solicitudes	68
4.7.9. Detalles de solicitud	69
4.8. Casos de uso	72
4.8.1. Actores	72
4.8.2. Antecedentes: Ingreso en la aplicación	73
4.8.3. Añadir experiencia laboral	75
4.8.4. Modificar experiencia laboral	77
4.8.5. Eliminar experiencia laboral	77
4.8.6. Búsqueda de experiencias laborales de otros usuarios	78
4.8.7. Solicitud de información privada de experiencias laborales a otros usuarios	81
4.8.8. Negociación sobre solicitudes de información privada	83

5. Pruebas y validación	89
5.1. Pruebas manuales	89
5.2. Pruebas automáticas de aceptación	89
5.2.1. Pruebas sobre experiencias laborales sin solicitudes	90
5.2.2. Pruebas sobre experiencias laborales con solicitudes aprobadas	91
6. Conclusiones	95
6.1. Consecución de objetivos	95
6.2. Lecciones aprendidas	96
6.3. Trabajos futuros	97
6.4. Código y estadísticas	98
A. Mockups del diseño inicial	101
Bibliografía	115

Índice de figuras

2.1. Diagrama de Gantt del análisis/diseño de la aplicación.	10
2.2. Diagrama de Gantt del desarrollo de la aplicación.	11
4.1. Esquema de la arquitectura general.	24
4.2. Representación de los objetos de valor del modelo de dominio.	25
4.3. Representación de las entidades del modelo de dominio.	26
4.4. Diagrama de entidad relación del modelo de datos.	31
4.5. Ejemplo de componente web <i>ActionDisplay</i>	44
4.6. Componente web <i>AppRouter</i>	45
4.7. Ejemplo de componente web <i>Autocomplete</i>	45
4.8. Ejemplo de componente web <i>AutocompleteMultiple</i>	46
4.9. Ejemplo de componente web <i>CardSkeleton</i>	47
4.10. Ejemplos de componentes web <i>Checkbox</i>	48
4.11. Ejemplo con varios componentes <i>Chip</i> seguidos.	48
4.12. Ejemplo con varios componentes <i>ClosableChip</i> seguidos.	49
4.13. Ejemplo de componente web <i>DateInput</i>	49
4.14. Ejemplo de componente web <i>Modal</i>	50
4.15. Ejemplo de componente web <i>MoneyInput</i>	51
4.16. Ejemplo de componente web <i>NavbarUser</i>	51
4.17. Ejemplo de componente web <i>NegotiableWorkExperience</i>	52
4.18. Ejemplo de componente web <i>NegotiationSummary</i>	53
4.19. Ejemplo de componente web <i>OwnWorkExperience</i>	54
4.20. Ejemplo de componente web <i>Radio</i>	54
4.21. Ejemplo de componente web <i>RequestButton</i>	55

4.22. Ejemplo de componente web <i>Select</i>	56
4.23. Ejemplo de componente web <i>WorkExperience</i>	56
4.24. Ejemplo de componente web <i>WorkExperienceSummary</i>	57
4.25. Diagrama de navegación entre las vistas de la aplicación.	59
4.26. Vista de inicio de sesión.	60
4.27. Vista de bienvenida.	61
4.28. Vista <i>Mis experiencias</i>	62
4.29. Vista <i>Mis experiencias</i> vacía.	62
4.30. Vista <i>Añadir experiencia</i>	64
4.31. Vista <i>Editar experiencia</i>	65
4.32. Vista <i>Buscar experiencias</i>	67
4.33. Vista <i>Solicitud de información</i>	68
4.34. Vista <i>Mis solicitudes</i> vacía.	69
4.35. Vista <i>Mis solicitudes</i>	70
4.36. Vista <i>Detalles de solicitud</i> durante la negociación.	71
4.37. Vista <i>Detalles de solicitud</i> al modificar las condiciones de la solicitud.	71
4.38. Vista <i>Detalles de solicitud</i> al concluir la negociación.	72
4.39. Casos de uso de la aplicación.	73
4.40. Página de inicio de sesión de GitHub.	74
4.41. Vista de bienvenida de la aplicación.	74
4.42. El <i>Usuario 1</i> está en la vista <i>Mis experiencias</i>	75
4.43. El <i>Usuario 1</i> rellena el formulario con la información de la experiencia.	76
4.44. El <i>Usuario 1</i> ve su experiencia laboral añadida.	76
4.45. El <i>Usuario 1</i> modifica los campos de su experiencia.	77
4.46. El <i>Usuario 1</i> ve su experiencia laboral modificada.	78
4.47. El <i>Usuario 1</i> confirma el borrado de su experiencia.	79
4.48. El <i>Usuario 1</i> ve que su experiencia laboral ha sido borrada.	79
4.49. El <i>Usuario 1</i> está en la vista <i>Buscar experiencias</i>	80
4.50. El <i>Usuario 1</i> filtra la búsqueda.	80
4.51. El <i>Usuario 1</i> está en la vista <i>Buscar experiencias</i>	81
4.52. El <i>Usuario 1</i> está en la vista <i>Solicitar información</i>	82

4.53. El <i>Usuario 1</i> está en la vista <i>Mis solicitudes</i>	82
4.54. El <i>Usuario 1</i> está en la vista <i>Detalles de solicitud</i>	83
4.55. El <i>Usuario 2</i> está en la vista <i>Mis solicitudes</i>	84
4.56. El <i>Usuario 2</i> está en la vista <i>Detalles de solicitud</i>	84
4.57. El <i>Usuario 2</i> negocia los términos de la solicitud en la ventana modal.	85
4.58. El estado de la solicitud cambia a pendiente de la respuesta del <i>Usuario 1</i>	85
4.59. El <i>Usuario 1</i> visualiza el cambio en el histórico de la solicitud.	86
4.60. El <i>Usuario 1</i> acepta la solicitud y puede ver los campos de las experiencias desvelados.	86
A.1. Vista inicial	102
A.2. Vista inicio de sesión	103
A.3. Vista registro básico	104
A.4. Vista registro completado	105
A.5. Vista home	106
A.6. Vista menú desplegable	107
A.7. Vista búsqueda de perfiles	108
A.8. Vista resultados de perfiles	109
A.9. Vista nueva experiencia laboral	110
A.10. Vista solicitud de negociación	111
A.11. Vista solicitudes pendientes	112
A.12. Vista información de perfiles laborales adquiridos	113
A.13. Vista detalles de información laboral adquirida	114

Capítulo 1

Introducción

Una de las dificultades a las que se enfrentan los alumnos en período de prácticas empresariales o recién egresados de la universidad es el desconocimiento de qué salario podría considerarse justo para un determinado puesto ante una oferta de trabajo.

A esta dificultad se añade el hecho de que muchos trabajadores consideran hablar de su sueldo un tema tabú, lo que complica a aquellos alumnos que buscan trabajo la tarea de recabar información acerca de qué rango salarial es el adecuado para una determinada actividad. Esta actitud generalizada puede beneficiar a las empresas que desean evitar conversaciones incómodas entre empleados que obtienen distintos salarios por desempeñar las mismas funciones [17, 43], pero no es beneficiosa para los demandantes de empleo.

Es habitual también que las propias empresas no publiquen información salarial en sus propias ofertas, por diversos motivos [37], entre los que se encuentra tener la capacidad durante el proceso de contratación de un empleado de ofrecer un salario que mejore al actual, pero que no tiene por qué estar alineado con el adecuado para su profesión.

Además, concretamente en el ámbito de las tecnologías de la información y la comunicación, el gran abanico de sectores, profesiones, técnicas, lenguajes, etcétera, hace que resulte realmente complejo para una persona con poca experiencia evaluar si una oferta de trabajo es interesante desde un punto de vista económico.

Con el objetivo de facilitar a los alumnos esta tarea de entendimiento del mercado, este proyecto pretende plantear una solución sencilla y eficaz en forma de aplicación web que permite buscar y compartir información laboral de primera mano a través de las experiencias laborales de los usuarios que participan en la plataforma.

Esta aplicación enfatiza en el intercambio de información laboral entre dos usuarios como fuente información, permitiendo en todos los casos que estos compartan únicamente los datos que deseen mostrar acerca de una experiencia laboral propia, incluida su identidad en caso de que prefieran ofrecer sus referencias de forma anónima.

Este proyecto se compone de un servidor que expone su funcionalidad a través de una API REST, desarrollado con la tecnología Spring Boot y una aplicación de navegador en el lado del cliente desarrollada en React que consume los recursos expuestos en dicha API. Todo el sistema es accesible a través de autenticación de los usuarios en la plataforma de GitHub, lo que permite a la mayoría de los alumnos de grados relacionados con las TIC poder hacer uso de la aplicación sin tener que hacer un registro adicional.

1.1. Definición de experiencia laboral

En el contexto de este Trabajo Fin de Grado, se considerará una experiencia laboral de un usuario al conjunto de datos formados por:

- La empresa donde transcurrió la experiencia laboral.
- El puesto que se desempeñó dentro de la empresa.
- El conjunto de tecnologías de las que el usuario hizo uso durante el ejercicio de su actividad.
- El periodo durante el que se ejerció la actividad en la empresa, con fecha de inicio y fecha de fin en el caso de experiencias pasadas.
- El salario bruto anual representativo que percibió el usuario durante el periodo laboral.

1.2. Privacidad

Para cada una de las experiencias que un usuario agregue al sistema, este podrá decidir todo momento la visibilidad que tendrá cada uno de los campos de la definición de experiencia laboral para el resto de usuarios, siendo posible que un campo sea público, lo que lo hará visible para la totalidad de los usuarios, o privado, en cuyo caso solo será visible para aquellos

usuarios que mediante una solicitud de información hayan obtenido permiso para visualizar dicho campo.

Adicionalmente, los usuarios podrán elegir dentro de sus experiencias, cuales de ellas serán vinculadas a su perfil. Aquellas que no sean vinculadas aparecerán vinculadas a un usuario anónimo en las búsquedas que hagan el resto de usuarios, siendo imposible reconocer a quién pertenecen.

1.3. Solicitud de información a otros usuarios

Dada una experiencia laboral determinada, en la que uno o varios de los campos hayan sido declarados como privados por su poseedor, es posible para otros usuarios solicitar revelar estos campos. Para esto el usuario interesado envía una solicitud de información en la que indica cuales de estos campos quiere que le sean revelados, y opcionalmente, ofrece revelar campos privados de alguna de sus experiencias a cambio al usuario receptor de la solicitud. En este momento se establece una negociación entre ambos usuarios, en el que cada uno de ellos, por turnos, puede modificar lo que ofrece y solicita revelar. Esta negociación finalizada cuando el usuario al que corresponde realizar el siguiente paso de la negociación acepta el intercambio de datos, o en el momento en el que uno de ambos usuarios deniega la solicitud.

1.4. Estado del arte

Antes de realizar este Trabajo Fin de Grado se evaluaron algunas herramientas que realizan una función parecida a la que la aplicación de este proyecto provee, entre ellas:

- Payscale.com [7]
- Salary.com [8]
- SalaryExpert.com [9]
- GlassDoor.com [6]

Estas herramientas tienen en común que la información laboral que recibe el usuario final proviene de la propia plataforma y no de primera mano, fruto de la interacción con otros usuarios, como se propone en este proyecto.

Cabe destacar también que existe otro Trabajo Fin de Grado [44] realizado en la Universidad Rey Juan Carlos que partió de la misma idea que fue propuesta para este TFG en 2016 por el mismo tutor, y que tuvo como resultado una aplicación móvil que almacenaba la mayoría de los datos en el propio dispositivo. En el planteamiento para el proyecto actual el almacenamiento de los datos laborales recae en el servidor, y la interfaz de usuario se propone como una aplicación web ejecutable en el navegador.

1.5. Estructura de la memoria

La organización de la información en esta memoria es la siguiente:

1. **Capítulo 1:** En este capítulo se exponen de forma introductoria la motivación y principales conceptos de este Trabajo Fin de Grado.
2. **Capítulo 2:** En este capítulo se describen los objetivos, requisitos y planificación temporal del proyecto.
3. **Capítulo 3:** En este capítulo se describen las tecnologías y herramientas utilizadas para el desarrollo de la aplicación, así como algunos conceptos teóricos de la arquitectura utilizada.
4. **Capítulo 4:** En este capítulo se explican los detalles de la implementación de la solución y su funcionalidad.
5. **Capítulo 5:** En este capítulo se describen las pruebas utilizadas para validar la aplicación.
6. **Capítulo 6:** En este capítulo se desarrollan las conclusiones obtenidas, los futuros trabajos y se referencia el código fuente del proyecto.

Capítulo 2

Objetivos

2.1. Objetivo general

El objetivo de este Trabajo Fin de Grado es desarrollar una aplicación que proporcione a estudiantes y trabajadores del sector de las tecnologías de la información y la comunicación un modo de evaluar cuál es el rango salarial justo para una determinada profesión u oferta de trabajo teniendo en cuenta la empresa donde se realizará la labor, el puesto de trabajo y las tecnologías relacionadas con el puesto mediante el intercambio de experiencias laborales entre usuarios, permitiendo a éstos mantener el control sobre la privacidad de sus datos y de su identidad.

2.2. Objetivos específicos

Los objetivos específicos de este Trabajo Fin de Grado son:

1. Proporcionar a los usuarios una plataforma donde compartir sus experiencias laborales.
2. Hacer esta plataforma fácilmente accesible para usuarios del sector TIC.
3. Permitir a los usuarios tener total control sobre la información laboral que comparten.
4. Permitir a los usuarios operar en la plataforma de forma anónima.
5. Proporcionar a los usuarios un mecanismo de búsqueda de experiencias laborales de su interés.
6. Proporcionar un mecanismo sencillo y atractivo para intercambiar información laboral.

2.3. Requisitos

Los requisitos que cumple la aplicación resultado de este Trabajo Fin de Grado son los siguientes:

1. Un usuario que tenga cuenta en GitHub podrá ingresar en la aplicación sin necesidad de registro.
2. La información de los usuarios (identificador, nombre de usuario y avatar) será recuperada de GitHub.
3. La información de usuario que ya se ha recuperado de GitHub no se volverá a solicitar (No funcional).
4. El usuario podrá crear un número indefinido de experiencias laborales a través de un formulario.
5. Las experiencias laborales creadas tendrán los siguientes campos:
 - Puesto de trabajo (Texto obligatorio).
 - Nombre de empresa (Texto obligatorio).
 - Tecnologías (Lista obligatoria de texto).
 - Salario (Cantidad económica obligatoria).
 - Fecha de inicio (Fecha pasada obligatoria).
 - Fecha de fin (Fecha pasada opcional).
6. Las experiencias laborales tendrán una visibilidad que puede ser pública o privada asociada a cada uno de los campos.
7. Las experiencias laborales tendrán una marca que indica si la experiencia laboral está vinculada al perfil del usuario o es anónima.
8. Un usuario podrá acceder a toda la información de sus propias experiencias laborales, sea pública a privada.
9. El usuario dispondrá de una vista donde se listarán sus experiencias laborales.

10. El usuario podrá eliminar cada una de sus experiencias laborales.
11. El usuario podrá editar todos los datos de sus experiencias laborales a través de un formulario.
12. El usuario dispondrá de una vista para buscar experiencias laborales de otros usuarios.
13. El usuario podrá filtrar experiencias en la búsqueda por los siguientes campos:
 - Puesto de trabajo: si se introduce cadena de caracteres en el filtro, permite obtener las experiencias cuyo puesto de trabajo sea público y contenga la cadena.
 - Nombre de empresa: si se introduce cadena de caracteres en el filtro, permite obtener las experiencias cuyo nombre de empresa sea público y contenga la cadena.
 - Tecnologías: si se introduce una serie de cadenas de caracteres en el filtro, permite obtener las experiencias cuyas tecnologías sean públicas y que cumplan que todos las cadenas del filtro son contenidas al menos por una de las tecnologías.
 - Salario mínimo: si se introduce un salario en el filtro, permite obtener las experiencias cuyo salario es superior al introducido.
 - Salario máximo: si se introduce un salario en el filtro, permite obtener las experiencias cuyo salario es inferior al introducido.
 - Fecha de entrada mínima: si se introduce una fecha en el filtro, permite obtener las experiencias cuya fecha de entrada es posterior a la introducida.
 - Fecha de entrada máxima: si se introduce una fecha en el filtro, permite obtener las experiencias cuya fecha de entrada es anterior a la introducida.
14. Los campos privados de una experiencia laboral ajena no se mostrarán a un usuario que consulta a menos que exista solicitud de experiencia laboral que cumpla todos estos criterios:
 - La solicitud está vinculada a dicha experiencia laboral.
 - La solicitud está vinculada además otra experiencia laboral del usuario que consulta.
 - La solicitud está en estado *aceptado*.

- Cuando se aceptó la solicitud, estaba establecido que se revelaría dicho campo de la experiencia.
15. La identidad del usuario al que pertenece una experiencia laboral ajena no se mostrará nunca a un usuario que consulta si esta está marcada como *no vinculada al perfil del usuario*, en su lugar el usuario se mostrará como *anónimo*.
16. Un usuario que tenga guardadas experiencias laborales podrá solicitar los campos privados de experiencias laborales ajenas a otros usuarios.
17. Un usuario podrá ofrecer campos privados de cualquiera de sus experiencias a la vez que solicita campos de experiencias laborales ajenas.
18. No se podrá negociar el ofrecimiento de la identidad de un usuario anónimo como el resto de campos.
19. El usuario tendrá una vista donde consultar aquellas solicitudes de información en las que participa, ya sea como solicitante o como receptor.
20. Una solicitud de información aparecerá al usuario en uno de los siguientes estados:
- Pendiente de su respuesta.
 - Pendiente de la respuesta de otro usuario.
 - Aceptada.
 - Cancelada.
21. Una solicitud de información contendrá referencias a la experiencia laboral ofrecida por el solicitante y a la experiencia laboral solicitada.
22. Una solicitud de información contendrá un histórico de acciones realizadas por los usuarios involucrados.
23. Una acción sobre una solicitud podrá ser de los siguientes tipos:
- Creación: Define qué campos se ofrece desbloquear tanto en la experiencia laboral ofrecida como en la solicitada.

- Modificación: Modifica los campos que se ofrece desbloquear.
 - Aceptación: Acepta la solicitud.
 - Cancelación: Cancela la solicitud.
24. Toda solicitud se crea con una acción de *creación* en su histórico.
25. Cualquier usuario involucrado en una solicitud podrá realizar una acción de *cancelación* en cualquier momento, a menos que esta solicitud se encuentre previamente aceptada o cancelada.
26. Un usuario podrá realizar una acción de *modificación* en una solicitud siempre que esta se encuentre en estado *pendiente de su respuesta*.
27. Una acción de *modificación* de una solicitud hace pasar el turno al otro usuario involucrado, es decir:
- Una vez realizada la acción por un usuario, este verá la solicitud en estado *pendiente de la respuesta de otro usuario*.
 - A la vez, la solicitud pasará al estado *pendiente de su respuesta* para el otro usuario.
28. Un usuario podrá realizar una acción de *aceptación* en una solicitud siempre que esta se encuentre en estado *pendiente de su respuesta*.
29. El usuario tendrá una vista donde para cada solicitud de información en la que esté envuelto podrá:
- Comprobar el estado de la solicitud.
 - Visualizar los datos del otro usuario envuelto en la solicitud, si este no aparece como anónimo por su relación con la experiencia laboral.
 - Visualizar la información de las experiencias laborales ofrecida y demandada en la solicitud.
 - Visualizar el histórico de acciones realizadas sobre la solicitud.
 - Cancelar la solicitud si aplica.
 - Aceptar la solicitud si aplica.

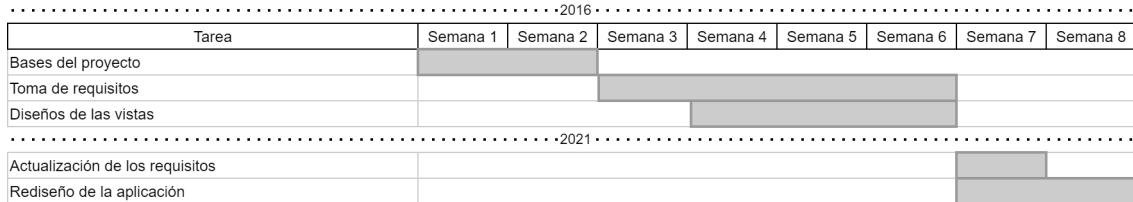


Figura 2.1: Diagrama de Gantt del análisis/diseño de la aplicación.

- Negociar la solicitud, añadiendo una acción de modificación, si aplica.
30. Una solicitud *aceptada* desvela los campos que se establecieron en la última acción de *creación/modificación* sobre ambas experiencias relacionadas a ambos usuarios involucrados.
31. El borrado de una experiencia laboral que estaba relacionada con una solicitud de información desencadenará el borrado de la solicitud independientemente de su estado y revocará sus efectos sobre la visibilidad de los campos que tienen los usuarios.

2.4. Planificación temporal

La idea para el desarrollo de este Trabajo Fin de Grado surgió en el año 2016. En esta primera fase la aplicación se concibió como una aplicación móvil, se establecieron las bases del proyecto y los requisitos funcionales y se realizaron los primeros diseños de la aplicación como *mockups* que pueden verse en el Apéndice A.

Después el proyecto se descontinuó hasta que se retomó en 2021, esta vez como una aplicación web.

El análisis y diseño de la aplicación a lo largo del tiempo teniendo en cuenta esta circunstancia se muestran en el diagrama de Gantt de la Figura 2.1.

El desarrollo de la aplicación a lo largo del tiempo se muestra en el diagrama de Gantt de la Figura 2.2.

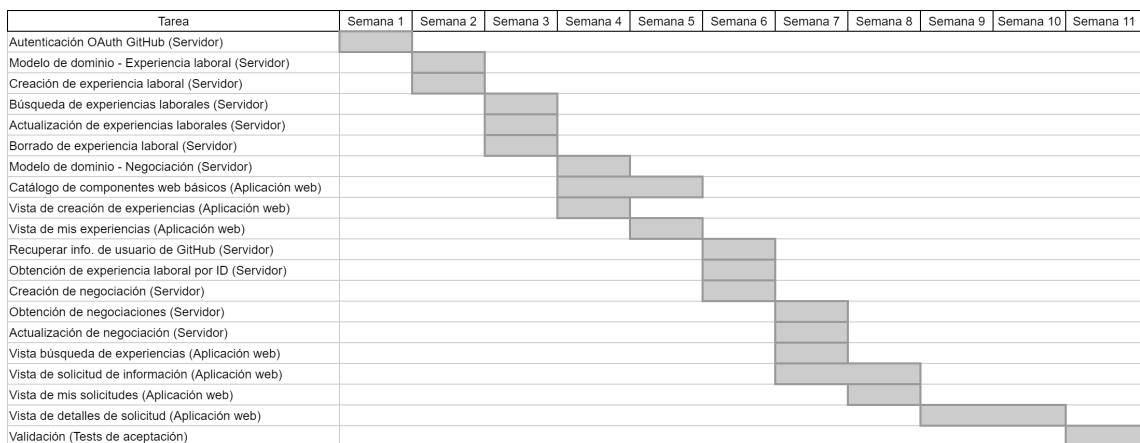


Figura 2.2: Diagrama de Gantt del desarrollo de la aplicación.

Capítulo 3

Tecnologías y herramientas

En este capítulo se describirán las tecnologías y herramientas empleadas para el desarrollo de la aplicación así como los principales aspectos teóricos de la arquitectura de software empleada.

3.1. Servidor de aplicaciones

Es la pieza de software que no se ejecuta en el navegador si no que es invocada desde este vía conexión HTTP [4]. Su cometido en la aplicación es el de exponer recursos que permiten al usuario consultar información y operar sobre ella [42], en este caso su experiencias laborales y solicitudes de información. Para ello antes de ofrecer información al lado cliente comprueba la identidad del usuario que la solicita para aplicarle las reglas de visibilidad y presentar únicamente la información que, o bien es pública, o bien pertenece al solicitante, o bien tanto el solicitante como el propietario han accedido a compartir. Abajo se describirán los puntos principales relacionados con el servidor de aplicaciones.

3.1.1. Lenguaje y plataforma

El lenguaje elegido para el desarrollo del servidor es Java [20]. Java es un lenguaje de programación orientada objetos [21] desarrollado por la compañía Sun Microsystems. Una de sus particularidades más importantes es la de poder ejecutarse en la máquina virtual de Java (JVM) [16] que puede ser instalada en la gran mayoría de sistemas operativos, lo que propor-

ciona a este lenguaje gran portabilidad entre distintas plataformas. La versión concreta de Java que se utilizó para el desarrollo fue la 11, que entre otras características da Java soporte para la programación funcional.

El framework Java utilizado para la creación del servidor fue Spring [34], un framework Open Source [19] que facilita el desarrollo de aplicaciones Java Enterprise Edition [14], que es a su vez una plataforma, parte de la plataforma Java que permite el desarrollo de aplicaciones servidor. Spring Framework provee una serie de características y librerías que ayudan al programador a centrarse en el desarrollo de la lógica que quiere implementar en su aplicación, abstrayéndose de conceptos de infraestructura que complican el desarrollo de servidores. Para esta aplicación se utilizó concretamente la tecnología Spring Boot que además reduce la cantidad de configuración necesaria para funcionamiento de una aplicación Spring.

3.1.2. Comunicación con el cliente

Para permitir a la parte cliente del navegador acceder la información laboral que reside en el servidor, la aplicación servidor expone una API REST [23]. Una API REST es una interfaz accesible de forma remota vía protocolo HTTP donde la información se representa en forma de recurso, siendo estos recursos identificadores de los conceptos y entidades que alberga el servidor, que en el caso de este Trabajo Fin de Grado son las experiencias laborales y las solicitudes de información. En una API REST estos recursos son operables utilizando los propios métodos HTTP, a los que el estilo REST provee de semántica, interpretándose los verbos HTTP como operaciones de escritura, lectura, borrado y modificación. De esta forma un cliente web puede en este caso crear, borrar, leer y modificar experiencias laborales y solicitudes dentro del servidor enviando mensajes HTTP.

3.1.3. Base de datos

Para almacenar las experiencias laborales de los usuarios y solicitudes, el servidor hace uso de una base de datos relacional (SQL) [12], concretamente H2 [30], que es un motor de bases de datos Open Source cuyas características principales son estar implementado en Java, lo que simplifica en muchos casos la integración con aplicaciones Java, y el poder levantarse en memoria o en modo embebido, de forma que sus datos se almacenan en ficheros, lo que facilita

la base de desarrollo, evitando al programador tener que instalar y configurar una base de datos completa. El dialecto SQL que utiliza H2 es altamente compatible con el resto de dialectos de SQL, lo que a futuro permite hacer uso de otro distribuidor distinto de base de datos sin realizar grandes modificaciones.

Como librería para realizar operaciones de base de datos desde el servidor se ha hecho uso de MyBatis [32], una herramienta de persistencia de Java que permite definir sentencias SQL en archivos XML [13] que pueden ser invocadas desde el código y que integra fácilmente con Spring Framework.

La herramienta para definir la estructura de la base de datos es Flyway [27], una tecnología que permite llevar un versionado de la base de datos mediante una lista de scripts SQL, uno por versión. Estos scripts se ejecutan secuencialmente, por lo que cada script debe escribirse como una modificación del estado de la estructura de base de datos que dejó el script anterior. De esta manera Flyway es capaz de llevar un registro de que modificaciones se hicieron en anteriores desarrollos sobre el proyecto, para ejecutar solo los nuevos scripts, es decir, los últimos cambios, no teniendo que generar una estructura nueva en cada modificación, lo que facilita la modificación de la base de datos una vez la aplicación está en producción, sin la pérdida de datos que supondría eliminar toda la estructura SQL y volverla a crear en cada modificación.

3.2. Aplicación web

Es la aplicación que se ejecuta en el navegador web del usuario [41]. Su función es la de ofrecer una interfaz amigable e intuitiva al usuario para permitirle operar sobre sus recursos en el servidor y realizar consultas de datos. En el caso concreto de este Trabajo Fin de Grado, la aplicación web ofrece una interfaz para que el usuario pueda crear, modificar e eliminar sus experiencias laborales, realizar búsquedas de experiencias laborales mediante filtros y crear solicitudes de datos confidenciales a otros usuarios desde su navegador, mediante una interfaz gráfica.

3.2.1. Lenguaje y librerías

El lenguaje utilizado para desarrollar la lógica en esta parte fue JavaScript [15]. Este es un lenguaje interpretado orientado a objetos, que se ejecuta en los navegadores y permite modificar

el documento HTML [11] que visualiza el usuario, solicitar datos remotos a un servidor y almacenar datos en el propio navegador entre otras funciones.

La librería utilizada para el desarrollo de la aplicación fue React [?], una librería que facilita el desarrollo de aplicaciones JavaScript de una sola página, cuya característica principal es la facilidad que aporta a la hora de desarrollar componentes web [40], es decir, componentes reutilizables en la aplicación, que se pueden invocar múltiples veces en el código mediante una etiqueta y atributos que actúan como entrada de información en el componente, y que tienen su propia estructura interna, comportamiento y estilo. Para esto React hace uso de archivos JavaScript con una extensión específica, JSX [18], que permiten referenciar HTML directamente desde el código JavaScript como si se tratara de un tipo de dato adicional a los que el JavaScript nativo soporta. Debido al uso de esta librería, la presencia de archivos HTML en la aplicación web es mínima, limitándose a un único archivo prácticamente vacío cuya función es referenciar los recursos necesarios para la ejecución de la aplicación.

3.2.2. Estilo

Para agregar estilos a la interfaz gráfica y hacer la aplicación visualmente más atractiva se ha hecho uso de la librería Bootstrap[25]. Bootstrap provee una serie de estilos CSS [1] predefinidos y fácilmente utilizables sobre los elementos HTML de la aplicación en forma de clases HTML. Además esta tecnología permite que las interfaces de usuario sean fácilmente adaptables a las pantallas de dispositivos móviles.

3.3. Autenticación

Con el objetivo de facilitar a los usuarios el ingreso en la aplicación, esta se ha integrado con la plataforma GitHub, haciendo uso de ella como proveedora de autenticación e identidad. De esta manera, los usuarios que ya están registrados en GitHub, como suele ser el caso de los alumnos de carreras relacionadas con las TIC, pueden utilizar sus credenciales de GitHub para acceder a la aplicación. Estas credenciales se envían directamente a GitHub, por lo que no pasan directamente por la aplicación de este Trabajo Fin de Grado, de forma que en ningún momento se pone en riesgo la privacidad del usuario.

Para lograr esto se ha hecho uso del módulo Spring Social [36], que facilita la comunicación

con GitHub [29] vía el estándar OAuth2 [22], una forma de integrar aplicaciones con plataformas que ofrecen parte de los datos de sus usuarios sin vulnerar su privacidad, extendiendo la sesión de estos usuarios a aplicaciones de terceros.

3.4. Herramientas

A continuación se exponen brevemente algunas de las principales herramientas que se utilizaron para el desarrollo de la aplicación.

3.4.1. IntelliJ IDEA

IntelliJ IDEA [31] es un entorno de desarrollo integrado desarrollado por la compañía JetBrains. Tiene soporte para el lenguaje Java, facilita las tareas de escritura del código, permite ejecutar y depurar aplicaciones y pruebas, integra con varias herramientas de gestión de dependencias y control de versiones, entre otras características. Se eligió este IDE para hacer el desarrollo del servidor de aplicaciones.

3.4.2. Visual Studio Code

Visual Studio Code [35] es un editor de código fuente con gran soporte para JavaScript. Permite ejecutar y depurar aplicaciones y pruebas, integra con gestores de dependencias y control de versiones. Se eligió este editor para el desarrollo de la aplicación web en React.

3.4.3. Maven

Maven [24] es una herramienta de gestión de proyectos de software desarrollada por Apache Software Foundation. Permite de forma declarativa gestionar todas las dependencias de un proyecto Java, es decir, referenciar los módulos que contienen las librerías sobre las que se construye el proyecto para poder descargarlos automáticamente del repositorio central de Maven durante la construcción de la aplicación. También permite la ejecución de plugins que facilitan tareas de desarrollo como construcción, ejecución de pruebas, empaquetado, instalación, etcétera. Se utilizó para la gestión de dependencias del servidor de aplicaciones.

3.4.4. NPM

NPM [33] es el sistema de gestión de paquetes para el lenguaje JavaScript. De forma similar a Maven permite gestionar las dependencias de una aplicación web JavaScript declarándolas en un fichero y automatizando de esta forma su descarga. Se utilizó para la gestión de dependencias de la aplicación web.

3.4.5. Git

Git [28] es un sistema de control de versiones distribuido y de código abierto. Permite mantener un histórico de los cambios del proyecto de forma remota (en este caso en GitHub) y una copia de estos de forma local. Además permite mantener distintas ramas de desarrollo en paralelo, esta característica facilita el desarrollo de distintas funcionalidades del software en paralelo, ya sea por uno o varios desarrolladores, siendo posible mezclar estas distintas ramas después con la versión final, gestionando mediante el sistema Git las diferencias y conflictos entre los cambios en el código.

3.5. Arquitectura hexagonal

La arquitectura hexagonal [10, 2], también denominada patrón de puertos y adaptadores, es un conjunto de patrones aplicados en el lado servidor de una aplicación que proporcionan una división en la aplicación en una serie de capas concéntricas con responsabilidades definidas. Esta división tiene como objetivo separar la lógica de negocio de una aplicación de las tecnologías utilizadas como consecuencia de la infraestructura en la que yace, es decir, bases de datos, buses de eventos, protocolos de transporte, etc. Algunas de las ventajas de esta arquitectura son:

- Las aplicaciones son más fáciles de probar mediante pruebas cuyo alcance es solo una de las capas en cada caso, falseando las demás en el contexto de la prueba.
- Las aplicaciones son adaptables ante posibles cambios en la infraestructura, ya que en estos casos la lógica de negocio se mantiene sin modificar mientras que la adaptación solo dependerá de implementar nuevos adaptadores para las nuevas tecnologías de infraestructura.

- Resulta más fácil por otra parte evolucionar y modificar la lógica de negocio teniendo en cuenta que esta es totalmente independiente de las tecnologías de infraestructura y frameworks.

En la arquitectura hexagonal intervienen las siguientes capas:

- Modelo de dominio [3]: Compuesto por las clases que representan los conceptos con los que el sistema opera. El modelo de dominio no debe tener dependencia con ninguna otra capa de la aplicación. Podemos distinguir las clases del modelo de dominio en dos tipos distintos:
 - Entidades [38]: son aquellas que son susceptibles de ser identificadas mediante un identificador único y que tienen ciclo de vida, es decir, evolucionan a lo largo del tiempo a través de las operaciones que los usuarios realizan sobre ellas. Podemos decir que dos entidades con igual información no son la misma entidad si difieren en su identificador único. Las entidades no son meras estructuras de datos anémicas, sino que contienen la lógica necesaria para construirse, para cambiar de estado, para validar la información de la que se componen o la que transitan y para interactuar con otras entidades. Además son las propias entidades las que generan proyecciones de si mismas omitiendo datos en caso de que el usuario que las accede no tenga visibilidad sobre estos datos, por lo que son las encargadas de aplicar los criterios de privacidad. Desde este punto de vista podemos afirmar que las entidades contienen la lógica de negocio de la aplicación.
 - Objetos de valor [39]: representan tipos de datos del dominio de las experiencias laborales que no tienen la capacidad de evolucionar a lo largo del tiempo, ya que de cambiar representarían realidades distintas. Se puede decir que dos objetos de valor con el mismo contenido son el mismo objeto de valor. Los objetos de valor tienen la capacidad de validarse en su construcción, de forma que no pueden construirse a partir de datos que no cumplan por ejemplo las restricciones de formato del objeto de valor.
- Capa de aplicación: Es la capa del sistema que expone una interfaz de servicios que pueden invocarse para realizar las distintas operaciones que componen la funcionalidad de

la aplicación. Además es la encargada de crear, persistir, recuperar y borrar las entidades del modelo de domino y de invocar su lógica de negocio. Es una aplicación Java pura, es decir, no requiere del framework, aunque puede utilizar librerías externas con un uso utilitario no requiere en este caso de contexto de Spring para funcionar, y solo depende de las clases del modelo de dominio. De este modo se logra que la lógica de negocio de la aplicación sea independiente de la infraestructura (protocolos de transporte, bases de datos, etcétera), haciendo el sistema fácilmente sea fácilmente acoplable a otras infraestructuras en caso de necesidad. Además permite realizar pruebas automáticas de lógica de negocio sin tener que levantar la infraestructura de la aplicación completa, de manera que son más descriptivas del comportamiento de la aplicación y más fáciles de mantener. Consta de los siguientes elementos:

- Interfaces de servicio: conocidas en la arquitectura como puertos primarios, son la entrada a la aplicación y representan el conjunto de operaciones que provee. Los métodos de estas interfaces aceptan objetos denominados comandos en el caso de las operaciones de escritura, y denominados consultas en el caso de las operaciones de lectura. Las consultas y comandos contienen los objetos de valor necesarios para la ejecución del método de servicio al que se pasan como parámetros.
- Implementación de los servicios: Implementa la anterior interfaz, por lo que tiene el código necesario para ejecutar la lógica de las operaciones desde la entrada del comando o consulta hasta la invocación de la interfaz de repositorio, con la consiguiente vuelta de información.
- Interfaces de repositorio: conocidas en la arquitectura como puertos secundarios, proveen a la aplicación del juego de métodos para operar contra los sistemas donde se persisten las entidades, sean estas del dominio del sistema o no. Las implementaciones de las interfaces de repositorio no son objeto de la capa aplicación, por estar fuertemente acopladas a tecnologías de transporte y persistencia, por lo que pertenecen a la siguiente capa, la capa de infraestructura.
- Capa de infraestructura: Es la capa del sistema que contiene el código que no está relacionado con la lógica de negocio sino con tecnologías de transporte y persistencia en base de datos, configuración y ejecución de la aplicación.

Capítulo 4

Diseño e implementación

En este capítulo se describirá con detalle técnico el diseño e implementación de la aplicación completa desarrollada para este Trabajo Fin de Grado.

4.1. Guía de lectura

La información técnica de esta sección está ordenada de la siguiente manera:

- Arquitectura general: Introduce al lector en la estructura del sistema desde un punto de vista global.
- Modelo de dominio: Se enfoca la implementación del modelo de dominio del servidor de la aplicación, donde reside la lógica de negocio y principales estructuras de datos.
- Modelo de datos: Describe la estructura de la base de datos del servidor.
- Diseño de la API REST: Describe los puntos de acceso que el servidor expone para invocar su funcionalidad.
- Componentes web: Se describe el catálogo de componentes desarrollado para su uso en la aplicación web.
- Vistas: Se describe de forma estática cada una de las pantallas que componen la aplicación web haciendo foco en el propósito funcional de la pantalla.

- Casos de uso: Se describen de forma dinámica las funcionalidades que provee la aplicación, con una guía que incluye figuras que representan el camino del usuario

Por ello se recomienda al lector que quiera empezar por la comprensión de la aplicación desde un punto de vista funcional que comience leyendo las secciones de Vistas 4.7 y Casos de uso 4.8.

4.2. Arquitectura general

Como se indicó anteriormente, el sistema está compuesto por una aplicación web de navegador desarrollada en React y un servidor de aplicaciones desarrollado en Spring Boot.

La aplicación web consiste en una única página web compuesta de componentes web de primer nivel, que denominamos vistas, y que contienen a su vez por componentes web de segundo nivel. Los componentes de primer nivel, llamados vistas, renderizan pantallas completas de la aplicación de una sola página, y tienen cada uno asociada una ruta de navegador. Los componentes de segundo nivel, renderizan elementos web reutilizables dentro de las distintas vistas, como entradas de texto, tarjetas, etcétera. En algunos casos un componente de segundo nivel puede hacer uso de otros componentes de segundo nivel para extender su funcionalidad. Además, la aplicación tiene un componente web de enrutado que permite navegar entre las distintas vistas, implementado con la tecnología React Router, que consiste en una barra de navegación horizontal en la parte superior de la pantalla.

El servidor fue desarrollado siguiendo una arquitectura hexagonal. Podemos definir la aplicación de esta arquitectura al sistema implementado para este Trabajo Fin de Grado a partir de los elementos que la componen, más interno a más externo:

- Modelo de dominio:
 - Entidades: En esta aplicación el dominio corresponde principalmente con los conceptos de experiencia laboral y negociación (también llamada solicitud de información), por lo tanto las entidades de este sistema son las experiencias laborales y las negociaciones.
 - Objetos de valor: ejemplos de objetos de valor de nuestro sistema serían las empresas o las tecnologías, ya que podemos afirmar que dos empresas con distinto nombre

son dos empresas distintas desde el punto de vista del sistema, al igual que ocurre con las tecnologías.

- Capa de aplicación:

- Interfaces de servicio: ejemplos de servicios son el servicio de experiencias laborales y el servicio de negociaciones, por lo tanto la interfaz de servicios se compone de operaciones como creación, actualización, borrado y consulta de experiencias laborales, o creación, actualización, recuperación, aceptación o denegación de negociaciones.
- Interfaces de repositorio: ejemplos de repositorios del sistema son el de experiencias laborales y el de negociaciones, que operan contra la base de datos del sistema para persistir y recuperar estas entidades, o el repositorio de usuarios, que opera contra GitHub para recuperar información de usuarios vía conexión REST.

- Capa de infraestructura:

- Spring Boot: utiliza clases de configuración que instancian en el contexto de Spring los bean necesarios para la ejecución de la aplicación.
- Spring MVC: define clases de tipo controlador que exponen endpoints invocables vía peticiones HTTP, estas clases conforman la API REST de la aplicación. También son las encargadas de, una vez deserializadas las peticiones HTTP, generar los comandos o consultas que serán enviados en la invocación a los puertos primarios de la aplicación.
- Spring Social: mediante configuración, levanta el mecanismo necesario para redirigir a GitHub al usuario para ingresar en la aplicación, y una vez dentro permite acceder al contexto de seguridad de Spring para recuperar datos de la sesión como la identidad del usuario.
- Spring Template: con esta tecnología para realizar llamadas HTTP se implementa la interfaz de repositorio de usuarios, que realiza llamadas a GitHub.
- MyBatis: con esta tecnología para operar contra bases de datos SQL se implementan los repositorios de experiencias laborales y negociaciones.

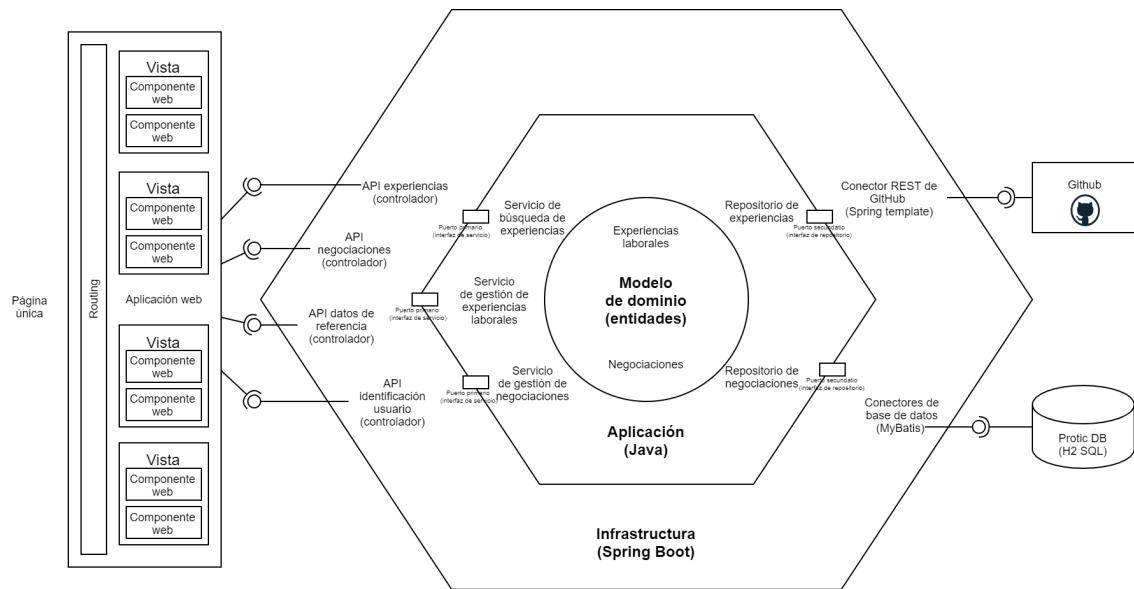


Figura 4.1: Esquema de la arquitectura general.

Para ilustrar el diseño de la arquitectura general de forma esquemática se incluye la Figura 4.1 que refleja los elementos descritos anteriormente.

4.3. Modelo de dominio

A continuación se describe la implementación del modelo de dominio de la aplicación. Para ello esta sección se estructurará en cinco subsecciones:

- Clases e interfaces comunes: aquellas clases del modelo de dominio comunes a toda la aplicación.
- Objetos de valor del dominio de las experiencias laborales: clases de los objetos de valor relacionados con la entidad experiencia laboral.
- Entidad experiencia laboral: descripción de las clases e interfaces que constituyen la entidad experiencia laboral.
- Objetos de valor del dominio de las negociaciones: clases de los objetos de valor relacionados con la entidad negociación.
- Entidad negociación: descripción de las clases e interfaces que constituyen la entidad negociación.

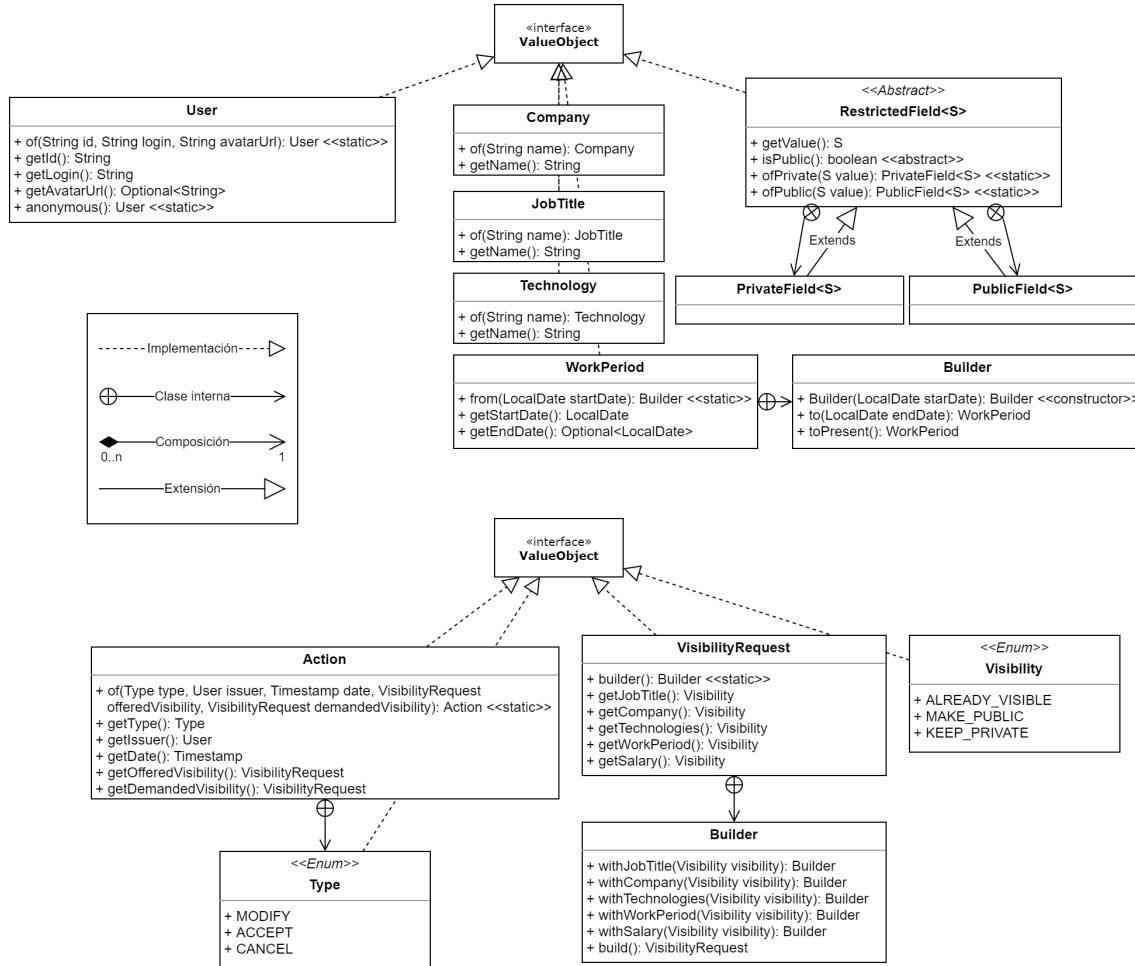


Figura 4.2: Representación de los objetos de valor del modelo de dominio.

4.3.1. Clases e interfaces comunes

Son las clases e interfaces utilizadas en todo el modelo de dominio de la aplicación. Se listan a continuación:

- **Identifiable**: Interfaz que implementan aquellas clases que se pueden identificar mediante un identificador único de tipo *UUID*.
- **TimeTraceable**: Interfaz que implementan aquellas clases que tienen una fecha de creación.
- **Entity**: Clase abstracta que extienden todas las entidades del modelo de dominio. Puesto que las entidades son siempre identificables y su fecha de creación es relevante, esta clase implementa ambas interfaces *Identifiable* y *TimeTraceable*

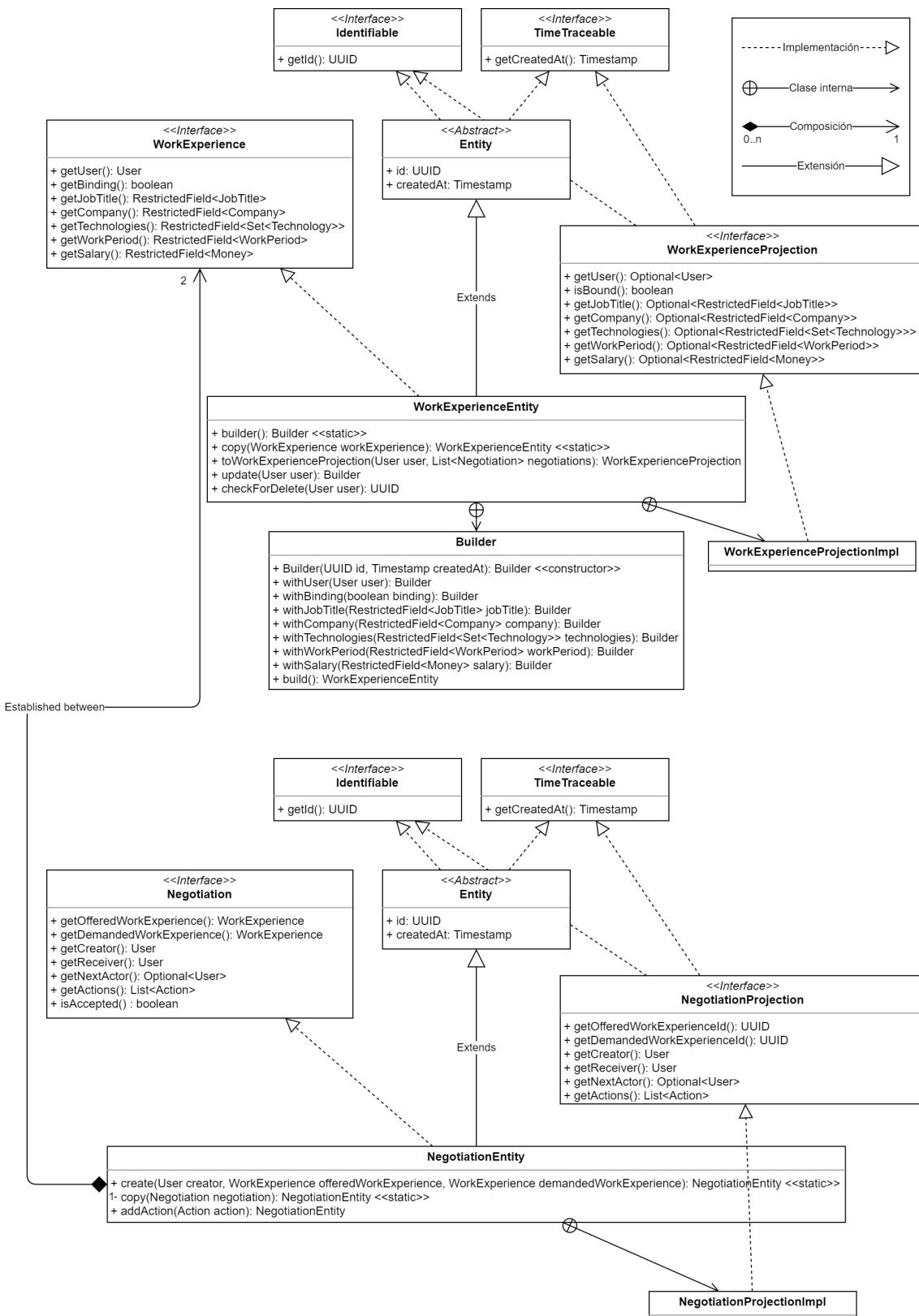


Figura 4.3: Representación de las entidades del modelo de dominio.

- **ValueObject:** Interfaz que implementan aquellas clases que son objetos de valor, es decir, aquellas clases que representan tipos de datos inmutables. Esta interfaz no tiene funcionalidad, su propósito es únicamente semántico.
- **User:** Objeto de valor que representa a un usuario de la aplicación. Contiene como datos el identificador del usuario en GitHub, su nombre de usuario y la URL de su avatar.

4.3.2. Objetos de valor del dominio de las experiencias laborales

Los objetos de valor del dominio de la experiencia laboral son clases que representan aquellos tipos de datos estáticos de los que se compone la información de la experiencia laboral. Su propósito es doble, por una parte aportan semántica al tipado de los atributos de la entidad experiencia laboral, por otra facilitan implementar las validaciones necesarias para construir estos atributos. Aparecen representados en la Figura 4.2. Se listan a continuación:

- **Company:** Representa la empresa donde tuvo lugar la experiencia laboral.
- **JobTitle:** Representa el puesto de trabajo de la experiencia laboral.
- **Technology:** Representa una tecnología que se manejó dentro de una experiencia laboral.
- **WorkPeriod:** Representa un periodo laboral con fecha de inicio y que puede tener o no fecha de fin. Se construye a través de una clase *Builder* que permite la creación de periodos laborales concluidos o que duran hasta la actualidad.
- **RestrictedField:** Es una clase genérica cuyo propósito es envolver a un objeto de valor de los tipos anteriores, añadiéndole en cada caso la visibilidad pública o privada. Para ello tiene dos extensiones llamadas *PrivateField* y *PublicField*. Se puede construir a partir de dos métodos estáticos que aceptan un objeto de valor como parámetro y devuelven un objeto de una de las dos extensiones que contiene al objeto de valor.

4.3.3. Entidad experiencia laboral

A continuación se describen las clases e interfaces que están relacionadas con el concepto de experiencia laboral como entidad de la aplicación, Aparecen representadas en la Figura 4.3.

- **WorkExperienceEntity:** Clase principal, representa directamente la entidad experiencia laboral. Se compone principalmente de una serie de atributos finales, estos atributos tienen como tipo uno de los objetos de valor de la experiencia laboral, pero envueltos como campos restringidos. Estos atributos, que pueden ser públicos o privados, son el nombre del puesto de trabajo, la empresa, la lista de tecnologías utilizadas, el periodo laboral y el salario. Adicionalmente esta entidad guarda la información del usuario al que pertenece, y su decisión de vincularla o no a su perfil. Esta entidad tiene en sus métodos la lógica de negocio necesaria para:
 - Actualizar su información comprobando previamente que el usuario que la actualiza es dueño de la entidad experiencia laboral que trata de actualizar.
 - Comprobar que un usuario que trata de borrar la entidad es dueño de esta para permitirlo o impedirlo en caso contrario.
 - Generar una proyección de la propia entidad para un usuario concreto que trata de consultarla, con los campos que el usuario solicitante no tiene permiso para ver omitidos.
- **WorkExperienceProjection:** Esta interfaz se devuelve en las consultas de experiencias laborales de la aplicación, en lugar de las propias entidades. Su utilidad es la de omitir todos aquellos atributos que el usuario solicitante no tiene permiso para ver. Para ello, la entidad genera proyecciones de este tipo siguiendo las siguientes reglas:
 - Un atributo siempre es visible para el poseedor de la experiencia laboral.
 - Un atributo es visible para cualquier usuario si es público.
 - Un atributo privado solo es visible para un usuario que no sea poseedor de la entidad, si existe para esta entidad alguna negociación entre el solicitante y el poseedor en estado aceptado que habilite el atributo.
 - El usuario poseedor de la entidad es siempre visible a sí mismo.
 - El usuario poseedor de la entidad es solo visible para otros usuarios si vinculó la experiencia laboral a su perfil. En caso contrario se mostrará como anónimo.
- **WorkExperience:** Esta interfaz es implementada por la entidad. Provee los métodos de

acceso a sus atributos, pero no contiene los métodos de negocio de la entidad. De esta forma se impide que fuera de la capa de aplicación se trate de invocar métodos de negocio.

4.3.4. Objetos de valor del dominio de las negociaciones

Los objetos de valor del dominio de las negociaciones son clases que representan aquellos tipos de datos estáticos que conforman una negociación. Su propósito es contener la información relativa al histórico de la negociación y la visibilidad que los usuarios ofrecen y solicitan en relación a las experiencias laborales. Aparecen representados en la Figura 4.2. Se listan a continuación:

- **Visibility:** Tipo enumerado que representa un cambio de visibilidad sobre un campo. Tiene tres posibles valores:
 - **MAKE_PUBLIC:** En relación a un campo de una experiencia laboral indica que se solicita que se convierta en público.
 - **KEEP_PRIVATE:** En relación a un campo de una experiencia laboral indica que no se solicita que el campo se convierta en público, sino que se mantenga privado.
 - **ALREADY_VISIBLE:** En relación a un campo de una experiencia laboral indica que el campo ya era público previamente a la negociación.
- **VisibilityRequest:** Representa una solicitud de visibilidad sobre una experiencia laboral. Contiene un atributo por campo de la experiencia laboral, de tipo *Visibility* indicando qué política de visibilidad se quiere establecer sobre el campo.
- **Action:** Representa una acción en el histórico de una negociación. Las acciones pueden ser de tipo *modificación*, *aceptación* y *cancelación* en función de si modifican la negociación o la aceptan o cancelan. Cada acción tiene una referencia al usuario que la ejecuta. Cada acción tiene ligadas la oferta de visibilidad sobre la experiencia laboral ofrecida y la solicitud de visibilidad sobre la oferta laboral solicitada, ambas del tipo *VisibilityRequest*.

4.3.5. Entidad negociación

A continuación se describen las clases e interfaces que están relacionadas con el concepto de negociación como entidad de la aplicación. Aparecen representados en la Figura 4.3.

- **NegotiationEntity:** Clase principal, representa directamente la entidad negociación. Se compone de:
 - La referencia al usuario creador de la negociación.
 - La referencia al usuario receptor de la negociación.
 - La experiencia laboral que el usuario creador de la negociación ofreció.
 - La experiencia laboral que el usuario creador de la negociación solicitó al usuario receptor.
 - El histórico de acciones que sucedieron durante la negociación.
 - La referencia al usuario que debe dar el siguiente paso en la negociación si esta está activa.

Esta entidad tiene en sus métodos la lógica de negocio necesaria para comprobar que un usuario que añade una acción a la negociación tiene permiso para hacerlo, comprobando si es el siguiente actor de la negociación.

- **NegotiationProjection:** Esta interfaz se devuelve en las consultas de negociaciones de la aplicación, en lugar de las propias entidades. Su utilidad es la de omitir todas las referencias de la negociación a un usuario cuando este no tiene vinculada con su perfil la experiencia laboral sobre la que se estable la negociación, de forma que el otro usuario no pueda conocer en ningún momento su identidad mientras negocia con él.
- **Negotiation:** Esta interfaz es implementada por la entidad. Provee los métodos de acceso a sus atributos, pero no contiene los métodos de negocio de la entidad. De esta forma se impide que fuera de la capa de aplicación se trate de invocar métodos de negocio.

4.4. Modelo de datos

Para explicar el modelo de datos de la aplicación se dará una breve descripción de cada una de las tablas que lo componen. La explicación se apoyará en el diagrama de la Figura 4.4

- **Tabla WORK EXPERIENCE:** En ella se almacenan los datos relativos a cada experiencia laboral perteneciente a los usuarios. Contiene el identificador único de la experiencia

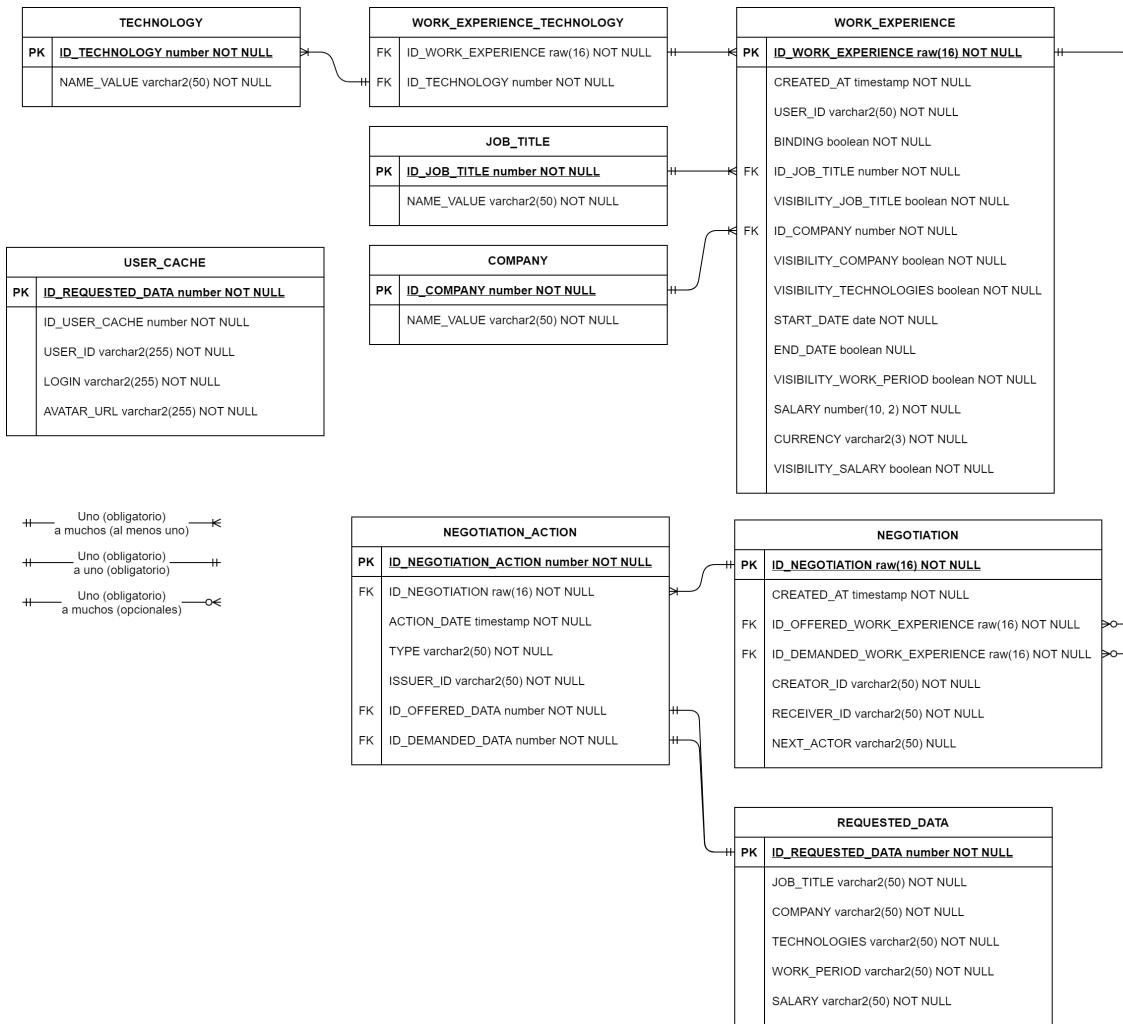


Figura 4.4: Diagrama de entidad relación del modelo de datos.

laboral, su fecha de creación, el identificador del usuario al que pertenece la experiencia, el indicador que marca si la experiencia está vinculada o no al perfil del usuario. Además, para cada campo de la experiencia laboral, esta tabla contiene:

- Puesto de trabajo: la visibilidad (privada o pública) del campo y una referencia a la tabla *JOB_TITLE* que contiene el puesto de trabajo.
 - Empresa: la visibilidad (privada o pública) del campo y una referencia a la tabla *COMPANY* que contiene el nombre de la empresa.
 - Tecnologías: la visibilidad (privada o pública) del campo. La tabla *WORK_EXPERIENCE_TECHNOLOGY* puede contener varias referencias a esta tabla, una por tecnología asociada a la experiencia laboral.
 - Período laboral: la visibilidad (privada o pública) del campo y las fechas de inicio y fin de la experiencia laboral, siendo la última nula si la experiencia dura hasta la actualidad.
 - Salario: la visibilidad (privada o pública) del campo y la cuantía del salario bruto anual, además del código ISO de la moneda.
-
- **Tabla JOB_TITLE:** Contiene los nombres de los puestos de trabajo relacionados con las experiencias laborales.
 - **Tabla COMPANY:** Contiene los nombres de las empresas relacionadas con las experiencias laborales.
 - **Tabla WORK_EXPERIENCE_TECHNOLOGY:** Relaciona las experiencias laborales con las tecnologías utilizadas en cada una de ellas, mediante una relación de muchas a muchas.
 - **Tabla TECHNOLOGY:** Contiene los nombres de las tecnologías relacionadas con las experiencias laborales.
 - **Tabla NEGOTIATION:** En ella se almacenan los datos relativos a las negociaciones. Contiene el identificador único de la negociación, su fecha de creación, una referencia

a la experiencia laboral ofrecida en la negociación y otra a la experiencia laboral solicitada, el identificador del usuario creador de la negociación y del usuario receptor, y el identificador del usuario que será el siguiente actor en la negociación, si aplica.

- **Tabla NEGOTIATION_ACTION:** En ella se almacena el histórico de acciones que han ido transcurriendo para cada negociación. Cada registro contiene la referencia a la negociación sobre la que se aplicó la acción, la fecha en la que se aplicó, el tipo de acción (modificación, aceptación o cancelación), el identificador del usuario que realizó la acción, y las referencias a la tabla *REQUESTED_DATA* donde se almacenan la visibilidad a aplicar a cada uno de los campos de la experiencia ofrecida y de la experiencia solicitada.
- **Tabla REQUESTED_DATA:** En ella se almacenan los posibles valores de la visibilidad de los campos para una experiencia laboral en negociación, siendo estos "manetener privado", "hacer público".^o za público".
- **Tabla USER_CACHE:** En esta tabla se almacena la información de los usuarios que se obtiene de GitHub, para evitar realizar peticiones HTTP para información de usuarios que ya se ha obtenido en el pasado.

4.5. Diseño de la API REST

En esta sección se describirá la API REST de la aplicación en cuatro grupos de puntos de API según su clasificación funcional:

- Puntos de API de sesión: aquellos relacionados con la sesión del usuario.
- Puntos de API de datos de referencia: aquellos que se utilizan para consultar puestos de trabajo, empresas y tecnologías que ya se dieron de alta en la aplicación.
- Puntos de API de experiencias laborales: aquellos relacionados con operaciones sobre la entidad experiencia laboral.
- Puntos de API de negociaciones: aquellos relacionados con operaciones sobre la entidad negociación.

4.5.1. Puntos de API de sesión

Es el conjunto de dos puntos de API que proveen operaciones relacionadas con la sesión del usuario:

Punto de API de obtención de usuario

Este punto de API se utiliza para obtener la información del usuario que encuentra en sesión. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */user*. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /user

-- Respuesta
200 OK
{
  "id": "1",
  "login": "mojombo",
  "avatarUrl": "https://avatars.githubusercontent.com/u/1?v=4"
}
```

Punto de API de cierre de sesión

Este punto de API se utiliza para invalidar la sesión del usuario en un cierre de sesión. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */user/logout*. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /user/logout

-- Respuesta
200 OK
```

4.5.2. Puntos de API de datos de referencia

Es el conjunto de tres puntos de API que se utilizan para proveer listados de información preexistente para las experiencias laborales. Se utilizan principalmente para alimentar componentes de tipo combo en la interfaz de usuario:

Punto de API de datos de puesto de trabajo

Este punto de API se utiliza para obtener la lista de nombres de puestos de trabajo que ya existen en la aplicación. Las solicitudes se envían mediante el método *GET* al recurso de la apli-

cación */data/job-titles*. La solicitud acepta un parámetro de consulta llamado *name* cuyo valor permite filtrar aquellos elementos de la lista que lo contienen en su nombre: A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /data/job-titles?name=pro

-- Respuesta
200 OK
{
  "data": [
    "ANALISTA/PROGRAMADOR",
    "DISEÑADOR DE PROCESOS",
    "PRODUCT OWNER",
    "PROGRAMADOR"
  ]
}
```

Punto de API de datos de empresas

Este punto de API se utiliza para obtener la lista de nombres de empresas que ya existen en la aplicación. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */data/companies*. La solicitud acepta un parámetro de consulta llamado *name* cuyo valor permite filtrar aquellos elementos de la lista que lo contienen en su nombre: A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /data/companies?name=in

-- Respuesta
200 OK
{
  "data": [
    "ARQUIMEA INGENIERÍA",
    "INDRA",
    "ING"
  ]
}
```

Punto de API de datos de tecnologías

Este punto de API se utiliza para obtener la lista de nombres de tecnologías que ya existen en la aplicación. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */data/technologies*. La solicitud acepta un parámetro de consulta llamado *name* cuyo valor permite filtrar aquellos elementos de la lista que lo contienen en su nombre: A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /data/technologies?name=j

-- Respuesta
200 OK
{
    "data": [
        "DJANGO",
        "JAVA",
        "JAVA 11",
        "JAVASCRIPT",
        "JQUERY",
        "VUE JS"
    ]
}
```

4.5.3. Puntos de API de experiencias laborales

Es el conjunto de dos puntos de API que proveen operaciones sobre la entidad experiencia laboral.

Punto de API de creación de experiencia laboral

Este punto de API se utiliza para crear una nueva experiencia laboral. Las solicitudes se envían mediante el método *POST* al recurso de la aplicación */work-experience*. El cuerpo de la solicitud contiene la información acerca de la experiencia laboral a crear en formato JSON. La respuesta contiene el identificador único de la nueva experiencia laboral. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
POST /work-experience
{
    "binding": true,
    "jobTitle": {
        "public": true,
        "content": "Analista/Desarrollador"
    },
    "company": {
        "public": true,
        "content": "Indra"
    },
    "technologies": {
        "public": true,
        "content": [
            "Java",
            "Spring"
        ]
    },
    "workPeriod": {
        "public": true,
        "content": {
            "startDate": "2020-04-10",
            "endDate": "2021-04-11"
        }
    },
    "salary": {
```

```

        "public": true,
        "content": {
            "value": 36000,
            "currency": "EUR"
        }
    }

-- Respuesta
200 OK
{
    "id": "99784c87-70d0-43cc-a340-aa28a19cb8a3"
}

```

Punto de API de obtención de experiencia laboral por identificador

Este punto de API se utiliza para recuperar una experiencia laboral por su identificador único. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */work-experience/{id}*, donde *id* es el identificador de la experiencia laboral a recuperar. La respuesta contiene los datos de la experiencia laboral, con los filtros de visibilidad aplicados según el usuario solicitante. A continuación se muestra un ejemplo de llamada al punto de API:

```

-- Solicitud
GET /work-experience/99784c87-70d0-43cc-a340-aa28a19cb8a3

-- Respuesta
200 OK
{
    "id": "99784c87-70d0-43cc-a340-aa28a19cb8a3",
    "user": {
        "id": "5689851",
        "login": "vicengg",
        "avatarUrl": "https://avatars.githubusercontent.com/u/5689851?v=4"
    },
    "binding": true,
    "jobTitle": {
        "content": "ANALISTA/DESARROLLADOR",
        "public": true
    },
    "company": {
        "content": "INDRA",
        "public": true
    },
    "technologies": {
        "content": [
            "JAVA",
            "SPRING"
        ],
        "public": true
    },
    "workPeriod": {
        "content": {
            "startDate": "2020-04-10",
            "endDate": "2021-04-11"
        },
        "public": true
    },
    "salary": {
        "content": {
            "value": 3.6E+4,
            "currency": "EUR"
        }
    }
}

```

```
        },
        "public": true
    }
}
```

Punto de API de búsqueda de experiencias laborales

Este punto de API se utiliza para buscar experiencias laborales filtrando por su contenido. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */work-experience*. La solicitud admite varios parámetros de consulta que filtran el resultado:

- *scope*: Indica si se solicitan todas las experiencias laborales (*scope=all*), las del usuario (*scope=own*) o las del resto de usuarios (*scope=foreign*). Su valor por defecto es *all*.
 - *jobTitle*: Al asignar un cadena de caracteres a este filtro, solo se devolveran aquellas experiencias que contengan esa en el nombre del puesto de trabajo.
 - *company*: Al asignar un cadena de caracteres a este filtro, solo se devolveran aquellas experiencias que contengan esa en el nombre de la empresa.
 - *technologies*: Al asignar múltiples cadenas de caracteres a este filtro, solo se devolveran aquellas experiencias que contengan esas cadenas en su lista de tecnologías.
 - *startDate*: Enviar este filtro hará que solo las experiencias laborales que iniciaron después de la fecha indicada estén presentes en la respuesta.
 - *endDate*: Enviar este filtro hará que solo las experiencias laborales que iniciaron antes de la fecha indicada estén presentes en la respuesta.
 - *minSalary*: Enviar este filtro hará que solo las experiencias laborales con un salario superior al indicado estén presentes en la respuesta.
 - *maxSalary*: Enviar este filtro hará que solo las experiencias laborales con un salario inferior al indicado estén presentes en la respuesta.

La respuesta contiene un listado de experiencias laborales, con los filtros de visibilidad aplicados según el usuario solicitante. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /work-experience?scope=own&jobTitle=programador&company=arquimea
&technologies=java&technologies=angular&startDate=2015-01-01
&endDate=2018-01-01&minSalary=5000&maxSalary=20000

-- Respuesta
200 OK
{
  "result": [
    {
      "id": "60219d70-ae4d-11eb-8529-0242ac130003",
      "user": {
        "id": "5689851",
        "login": "vicengg",
        "avatarUrl": "https://avatars.githubusercontent.com/u/5689851?v=4"
      },
      "binding": true,
      "jobTitle": {
        "content": "PROGRAMADOR",
        "public": true
      },
      "company": {
        "content": "ARQUIMEA INGENIERÍA",
        "public": true
      },
      "technologies": {
        "content": [
          "JAVASCRIPT",
          "JAVA",
          "SPRING",
          "JQUERY",
          "ANGULAR"
        ],
        "public": true
      },
      "workPeriod": {
        "content": {
          "startDate": "2015-03-01",
          "endDate": "2017-07-01"
        },
        "public": true
      },
      "salary": {
        "content": {
          "value": 1.1E+4,
          "currency": "EUR"
        },
        "public": false
      }
    }
  ]
}
```

Punto de API de actualización de experiencia laboral

Este punto de API se utiliza para actualizar una experiencia laboral. Las solicitudes se envían mediante el método *POST* al recurso de la aplicación */work-experience/{id}*, donde *id* es el identificador de la experiencia laboral a actualizar. El cuerpo de la solicitud contiene la información acerca de la experiencia laboral a actualizar en formato JSON. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
PUT /work-experience/99784c87-70d0-43cc-a340-aa28a19cb8a3
```

```
{
    "binding": true,
    "jobTitle": {
        "public": true,
        "content": "Analista/Desarrollador"
    },
    "company": {
        "public": true,
        "content": "Indra"
    },
    "technologies": {
        "public": true,
        "content": [
            "Java",
            "Spring"
        ]
    },
    "workPeriod": {
        "public": true,
        "content": {
            "startDate": "2020-04-10",
            "endDate": "2021-04-11"
        }
    },
    "salary": {
        "public": true,
        "content": {
            "value": 36000,
            "currency": "EUR"
        }
    }
}

-- Respuesta
200 OK
```

Punto de API de borrado de experiencia laboral

Este punto de API se utiliza para borrar una experiencia laboral. Las solicitudes se envían mediante el método *DELETE* al recurso de la aplicación */work-experience/{id}*, donde *id* es el identificador de la experiencia laboral a borrar. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
DELETE /work-experience/99784c87-70d0-43cc-a340-aa28a19cb8a3

-- Respuesta
200 OK
```

4.5.4. Puntos de API de negociaciones

Es el conjunto de dos puntos de API que proveen operaciones sobre la entidad negociación.

Punto de API de creación de negociación

Este punto de API se utiliza para crear una nueva negociación. Las solicitudes se envían mediante el método *POST* al recurso de la aplicación */negotiation*. El cuerpo de la solicitud contiene las referencias a las experiencias laborales entre las que se establece la negociación. La respuesta contiene el identificador único de la nueva negociación. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
POST /negotiation
{
    "offeredWorkExperienceId": "4c714c8e-9d5f-11eb-a8b3-0242ac130003",
    "demandedWorkExperienceId": "b6b3705e-9e04-11eb-a8b3-0242ac130003"
}

-- Respuesta
200 OK
{
    "id": "958961ae-deaa-484f-b3a7-39bcb7ca0df0"
}
```

Punto de API de agregación de acciones a una negociación

Este punto de API se utiliza para agregar una nueva acción a una negociación. Las solicitudes se envían mediante el método *PUT* al recurso de la aplicación */negotiation/{id}*, donde *id* es el identificador de la negociación a la que se le va a agregar una nueva acción. El cuerpo de la solicitud contiene la información de la nueva acción a añadir. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
PUT /negotiation/958961ae-deaa-484f-b3a7-39bcb7ca0df0/action
{
    "type": "accept",
    "offeredData": {
        "jobTitle": "make_public",
        "company": "make_public",
        "technologies": "make_public",
        "workPeriod": "make_public",
        "salary": "make_public"
    },
    "demandedData": {
        "jobTitle": "make_public",
        "company": "make_public",
        "technologies": "make_public",
        "workPeriod": "make_public",
        "salary": "make_public"
    }
}

-- Respuesta
200 OK
```

Punto de API de obtención de negociación por identificador

Este punto de API se utiliza para recuperar una negociación por su identificador único. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */negotiation/{id}*, donde *id* es el identificador de la negociación a recuperar. La respuesta contiene los datos de la negociación, con los filtros de visibilidad aplicados a algunos campos según el usuario solicitante. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /negotiation/958961ae-deaa-484f-b3a7-39bcb7ca0df0

-- Respuesta
200 OK
{
  "id": "958961ae-deaa-484f-b3a7-39bcb7ca0df0",
  "offeredWorkExperienceId": "4c714c8e-9d5f-11eb-a8b3-0242ac130003",
  "demandedWorkExperienceId": "b6b3705e-9e04-11eb-a8b3-0242ac130003",
  "creator": {
    "id": "5689851",
    "login": "vicengg",
    "avatarUrl": "https://avatars.githubusercontent.com/u/5689851?v=4"
  },
  "receiver": {
    "id": "0",
    "login": "anónimo",
    "avatarUrl": "https://crysteland.com/wp-content/uploads/2016/12/unknown-user-460x460.png"
  },
  "nextActor": {
    "id": "0",
    "login": "anónimo",
    "avatarUrl": "https://crysteland.com/wp-content/uploads/2016/12/unknown-user-460x460.png"
  },
  "actions": [
    {
      "type": "MODIFY",
      "date": "2021-05-05T13:59:15.229+00:00",
      "offeredData": {
        "jobTitle": "MAKE_PUBLIC",
        "company": "MAKE_PUBLIC",
        "technologies": "MAKE_PUBLIC",
        "workPeriod": "MAKE_PUBLIC",
        "salary": "MAKE_PUBLIC"
      },
      "demandedData": {
        "jobTitle": "MAKE_PUBLIC",
        "company": "MAKE_PUBLIC",
        "technologies": "MAKE_PUBLIC",
        "workPeriod": "MAKE_PUBLIC",
        "salary": "MAKE_PUBLIC"
      },
      "issuer": {
        "id": "5689851",
        "login": "vicengg",
        "avatarUrl": "https://avatars.githubusercontent.com/u/5689851?v=4"
      }
    }
  ]
}
```

Punto de API de obtención de negociaciones

Este punto de API se utiliza obtener las negociaciones en las cuales el usuario solicitantes está implicado. Las solicitudes se envían mediante el método *GET* al recurso de la aplicación */negotiation*. La solicitud admite un parámetro de consulta, *scope*, este parámetro indica si deben devolverse las negociaciones de las que el usuario es creador (*scope=creator*) o de las que el usuario es receptor (*scope=receiver*). Por defecto su valor es(*scope=creator*). La respuesta contiene un listado de negociaciones. A continuación se muestra un ejemplo de llamada al punto de API:

```
-- Solicitud
GET /work-experience?scope=own&jobTitle=programador&company=arquimea
&technologies=java&technologies=angular&startDate=2015-01-01
&endDate=2018-01-01&minSalary=5000&maxSalary=20000

-- Respuesta
200 OK
{
  "result": [ {
    "id": "958961ae-deaa-484f-b3a7-39bcb7ca0df0",
    "offeredWorkExperienceId": "4c714c8e-9d5f-11eb-a8b3-0242ac130003",
    "demandedWorkExperienceId": "b6b3705e-9e04-11eb-a8b3-0242ac130003",
    "creator": {
      "id": "5689851",
      "login": "vicengg",
      "avatarUrl": "https://avatars.githubusercontent.com/u/5689851?v=4"
    },
    "receiver": {
      "id": "0",
      "login": "anónimo",
      "avatarUrl": "https://crysteland.com/wp-content/uploads/2016/12/unknown-user-460x460.png"
    },
    "nextActor": {
      "id": "0",
      "login": "anónimo",
      "avatarUrl": "https://crysteland.com/wp-content/uploads/2016/12/unknown-user-460x460.png"
    },
    "actions": [ {
      "type": "MODIFY",
      "date": "2021-05-05T13:59:15.229+00:00",
      "offeredData": {
        "jobTitle": "MAKE_PUBLIC",
        "company": "MAKE_PUBLIC",
        "technologies": "MAKE_PUBLIC",
        "workPeriod": "MAKE_PUBLIC",
        "salary": "MAKE_PUBLIC"
      },
      "demandedData": {
        "jobTitle": "MAKE_PUBLIC",
        "company": "MAKE_PUBLIC",
        "technologies": "MAKE_PUBLIC",
        "workPeriod": "MAKE_PUBLIC",
        "salary": "MAKE_PUBLIC"
      },
      "issuer": {
        "id": "5689851",
        "login": "vicengg",
        "avatarUrl": "https://avatars.githubusercontent.com/u/5689851?v=4"
      }
    }]
  }]
}
```

The screenshot shows a component with two sections. The left section is for user 'vicengg' and the right section is for user 'vgimenezg'. Each section contains a user icon, the user's name, a message about the action being recent, and a list of items. The items listed are 'Tecnologías' and 'Periodo laboral'.

Usuario	Mensaje	Acciones
vicengg	creó la solicitud hace menos de 1 minuto.	Tecnologías Periodo laboral
vgimenezg	mostrará:	Tecnologías Periodo laboral

Figura 4.5: Ejemplo de componente web ActionDisplay.

}

4.6. Componentes web

En esta sección se describen los componentes web que se implementaron para su uso en la interfaz de usuario de la aplicación. Los componentes web utilizados para este Trabajo Fin de Grado no pertenecen a ninguna librería, sino que fueron desarrollados por el autor desde cero utilizando la librería de desarrollo React. Por ello lo que a continuación se describe es el catálogo completo desarrollado explicando propósito, interfaz de programación e interfaz gráfica de cada componente creado. Los estilos aplicados a cada componente sí provienen de la hoja de estilos de Bootstrap.

4.6.1. ActionDisplay

Este componente que se utiliza en la vista de detalles de una negociación se utiliza para visualizar cada una de las acciones del histórico de una negociación. Informa al usuario que realizó la acción, el tipo de acción que realizó, el tiempo que transcurrió desde que se realizó la acción y, en caso de que la acción sea una modificación de la negociación, la visibilidad que se ofrece y solicita entre las experiencias laborales. Este componente se muestra en la Figura 4.5.

Sus entradas son:

- **action:** Objeto con el contenido de la acción, ver definición de *Action* en la Subsección 4.3.4.
- **user:** Objeto que contiene la información del usuario en sesión, ver definición de *User* en la Subsección 4.3.1.

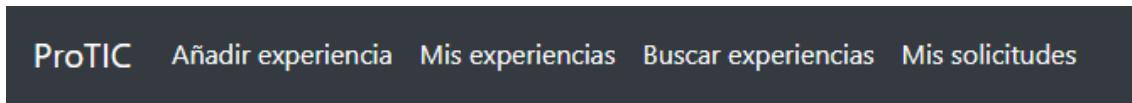


Figura 4.6: Componente web *AppRouter*.

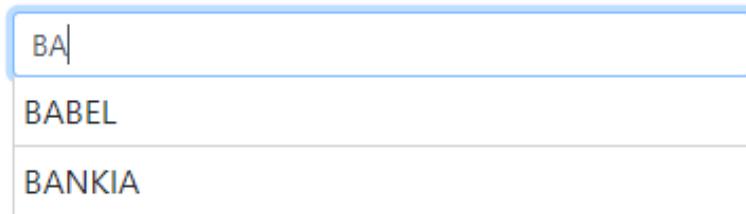


Figura 4.7: Ejemplo de componente web *Autocomplete*.

- **isFirst:** Indica si el *ActionDisplay* es o no el primer elemento de una lista, para su correcta visualización.
- **creator:** Objeto que contiene la información del usuario que creó la negociación, ver definición de *User* en la Subsección 4.3.1.
- **receiver:** Objeto que contiene la información del usuario que inició la negociación, ver definición de *User* en la Subsección 4.3.1.

4.6.2. AppRouter

Este componente establece la relación de rutas entre las distintas vistas de la aplicación y permite navegar a algunas de ellas mediante una barra de navegación. Este componente se muestra en la Figura 4.6.

4.6.3. Autocomplete

Este componente es una entrada de texto que ofrece al usuario una lista de sugerencias como ayuda para completar el texto que está ingresando. Los elementos aparecen filtrados según el texto que el usuario va escribiendo. Para ofrecer la lista de sugerencias este formulario realiza una petición HTTP a uno de los puntos de API definidos en la Subsección 4.5.2. Este componente se muestra en la Figura 4.7.

Sus entradas son:

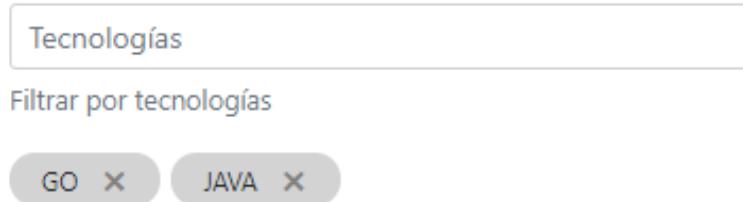


Figura 4.8: Ejemplo de componente web *AutocompleteMultiple*.

- **url:** URL de datos de referencia a la que consulta el componente, ver Subsección 4.5.2.
- **placeholder:** Texto opcional que se muestra en la entrada de texto cuando el usuario no ha escrito nada.
- **footer:** Texto opcional que se muestra como pie de la entrada de texto.
- **value:** Variable que contiene el texto ingresado en la entrada de texto.
- **onChange:** Función de tipo callback que se ejecuta cuando hay cambios en la entrada de texto.
- **onSelect:** Función de tipo callback que se ejecuta cuando el usuario selecciona una sugerencia.
- **onSubmit:** Función de tipo callback que se ejecuta cuando el usuario presiona la tecla *introducir*.
- **styleClasses:** Clases adicionales de estilos que se pueden añadir al componente.

4.6.4. AutocompleteMultiple

Este componente es una entrada de texto que ofrece al usuario una lista de sugerencias como ayuda para completar el texto que está ingresando, similar al componente descrito anteriormente en la Subsubsección 4.6.3, pero que de forma diferente a este permite seleccionar múltiples valores que se muestran a continuación de la entrada de texto. Los elementos aparecen filtrados según el texto que el usuario va escribiendo. Para ofrecer la lista de sugerencias este formulario realiza una petición HTTP a uno de los puntos de API definidos en la Subsección 4.5.2. Este componente se muestra en la Figura 4.8.

Sus entradas son:

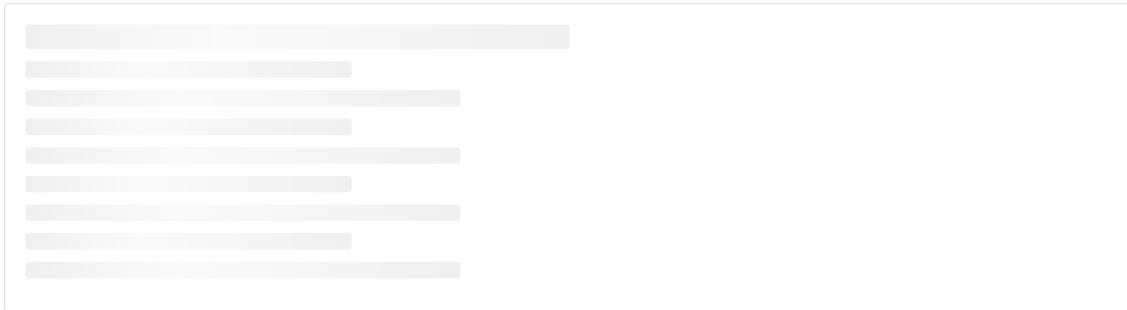


Figura 4.9: Ejemplo de componente web *CardSkeleton*.

- **url:** URL de datos de referencia a la que consulta el componente, ver Subsección 4.5.2.
- **placeholder:** Texto opcional que se muestra en la entrada de texto cuando el usuario no ha escrito nada.
- **footer:** Texto opcional que se muestra como pie de la entrada de texto.
- **values:** Variable que contiene las lista de valores ingresados en la entrada de texto.
- **setValues:** Función que establece el valor de la variable *values*.
- **styleClasses:** Clases adicionales de estilos que se pueden añadir al componente.

4.6.5. CardSkeleton

Componente visual que muestra una tarjeta con datos en blanco, utilizado para mantener una experiencia de usuario agradable durante el tiempo que tardan en cargarse algunos que se consulta a la API REST. Este componente se muestra en la Figura 4.9.

Sus entradas son:

- **lines:** Número de líneas en blanco que componen la tarjeta.

4.6.6. Checkbox

Este componente permite seleccionar un valor booleano para una variable mediante un elemento de tipo casilla de verificación, o un interruptor dependiendo del estilo que se elija. El componente se muestra en la Figura 4.10.

Sus entradas son:



Figura 4.10: Ejemplos de componentes web *Checkbox*.

NODE MICROSOFT AZURE JAVA 11 OPENSIFT SPRING BOOT

Figura 4.11: Ejemplo con varios componentes *Chip* seguidos.

- **value:** Variable que contiene el valor de la casilla o interruptor.
- **setValue:** Función que establece el valor de la variable *value* en los cambios de estado de la casilla o interruptor.
- **labelOff:** Texto que acompaña al elemento cuando está desmarcado.
- **labelOn:** Texto que acompaña al elemento cuando está marcado.
- **type:** Por defecto *checkbox*, muestra una casilla de verificación si su valor es *checkbox* o un interruptor si su valor es *switch*.
- **disabled:** Si se establece su valor a *true*, deshabilita el componente para que no sea modificable.

4.6.7. Chip

Componente de tipo visual que permite representar un texto envuelto en un óvalo. Se usa para representar un elemento en un listado de valores. Este componente se muestra en la Figura 4.11.

Sus entradas son:

- **value:** Texto del elemento de tipo *Chip*.

4.6.8. ClosableChip

Componente de tipo visual que permite representar un texto envuelto en un óvalo, similar al componente Chip 4.6.7, pero que a diferencia de este permite ser eliminado realizando click

DJANGO X DOCKER X JAVA 11 X MICROSOFT AZURE X

Figura 4.12: Ejemplo con varios componentes *ClosableChip* seguidos.

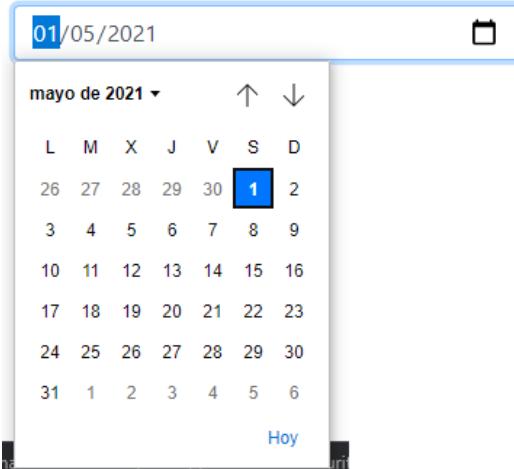


Figura 4.13: Ejemplo de componente web *DateInput*.

en un botón de aspa. Se usa para representar un elemento eliminable en un listado de valores.

Este componente se muestra en la Figura 4.12.

Sus entradas son:

- **value:** Texto del elemento de tipo *Chip*.
- **onClose:** Función que se ejecuta cuando se realiza click en el botón de aspa.

4.6.9. DateInput

Este componente es una entrada de texto que permite al usuario ingresar fechas. Este componente se muestra en la Figura 4.13.

Sus entradas son:

- **footer:** Texto opcional que se muestra como pie de la entrada de texto.
- **value:** Variable que contiene el texto ingresado en la entrada de texto.
- **onChange:** Función de tipo callback que se ejecuta cuando hay cambios en la entrada de texto.

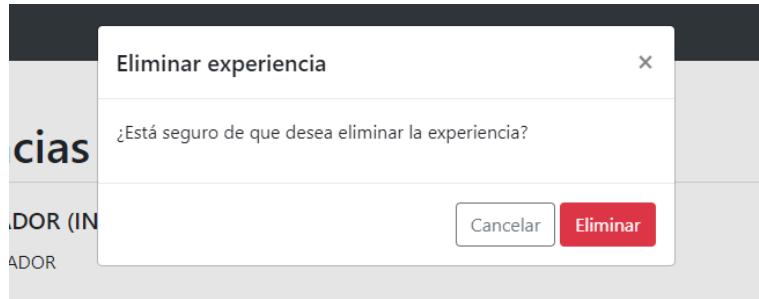


Figura 4.14: Ejemplo de componente web *Modal*.

- **disabled:** Si se establece a *true*, la entrada de texto permanece deshabilitada para que no sea modificable.
- **styleClasses:** Clases adicionales de estilos que se pueden añadir al componente.

4.6.10. Modal

Este componente web es una ventana modal que emerge en la página. Se utiliza para destacar operaciones que requieren atención especial por parte del usuario en el contexto de una vista. Este componente se muestra en la Figura 4.14.

Sus entradas son:

- **title:** Título de modal.
- **children:** Una lista de dos elementos HTML (o componentes web) que el modal puede mostrar como contenido, el primero en el cuerpo del modal y el segundo en el pie del modal.
- **isShown:** Variable que controla si el modal se muestra o no.
- **close:** Función que maneja el cierre del modal.
- **additionalClasses:** Clases adicionales de estilos que se pueden añadir al componente.

4.6.11. MoneyInput

Este componente es una entrada de texto que permite al usuario ingresar cantidades de dinero. Este componente se muestra en la Figura 4.15.

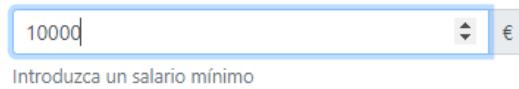


Figura 4.15: Ejemplo de componente web *MoneyInput*.

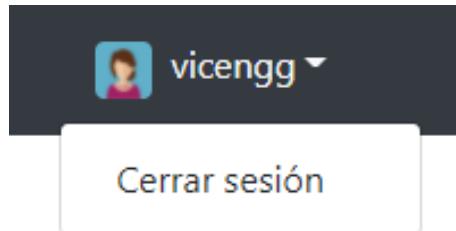


Figura 4.16: Ejemplo de componente web *NavbarUser*.

Sus entradas son:

- **placeholder:** Texto opcional que se muestra en la entrada de texto cuando el usuario no ha escrito nada.
- **footer:** Texto opcional que se muestra como pie de la entrada de texto.
- **value:** Variable que contiene el dinero ingresado en la entrada de texto.
- **onChange:** Función de tipo callback que se ejecuta cuando hay cambios en la entrada de texto.
- **currency:** Código ISO de la moneda en la cual se introduce la cantidad.
- **styleClasses:** Clases adicionales de estilos que se pueden añadir al componente.

4.6.12. NavbarUser

Este componente muestra el usuario que está en sesión en la barra superior de navegación. Además le permite cerrar sesión mediante un botón desplegable. Este componente consulta directamente al punto de API de obtención de usuario para recuperar los datos del usuario en sesión. Ver 4.5.1. Este componente se muestra en la Figura 4.16



Figura 4.17: Ejemplo de componente web *NegotiableWorkExperience*.

4.6.13. NegotiableWorkExperience

Este componente de tipo tarjeta permite operar sobre los campos de una experiencia laboral, modificando la visibilidad de los campos de cara a realizar una oferta o solicitud de información. Cada uno de los campos privados de la experiencia laboral posee su propio interruptor que controla la visibilidad, el usuario elige si quiere mantener un campo privado o ofrecerlo/pedirlo como público. Este componente se muestra en la Figura 4.17.

Sus entradas son:

- **workExperience:** Objeto con el contenido de la experiencia laboral sobre la que se va a hacer el ofrecimiento/solicitud de visibilidad, ver definición de *WorkExperienceEntity* en la Subsección 4.3.3.
- **children:** Lista de tres elementos HTML. La tarjeta puede contener hasta tres elementos HTML (o componentes web) en su cuerpo. Los dos primeros representados a la izquierda del avatar del usuario y el último justo debajo.
- **visibilityRequest:** Variable que contiene la información de la visibilidad de la experiencia

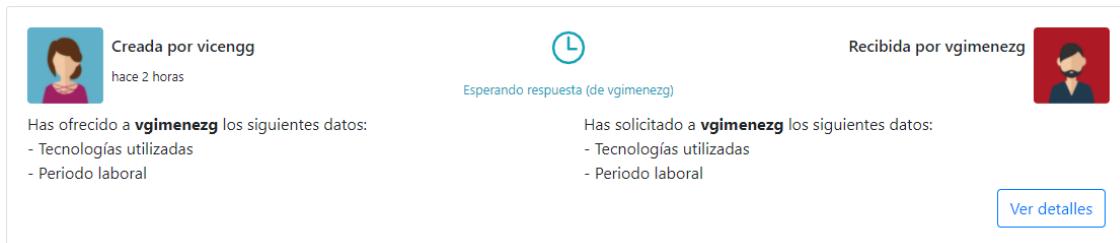


Figura 4.18: Ejemplo de componente web *NegotiationSummary*.

laboral que se está manipulando en el componente, ver definición de *VisibilityRequest* en la Subsección 4.3.4.

- **setVisibilityRequest:** Función que establece el valor de la variable *visibilityRequest*.
- **disabled:** Si se establece a *true* deshabilita el componente para evitar su modificación.

4.6.14. NegotiationSummary

Este componente de tipo tarjeta muestra de forma resumida el último estado de una negociación, es decir, si está pendiente, aceptada o cancelada, los datos que los usuarios ofrecen y solicitan y los usuarios que intervienen en ella. Este componente se muestra en la Figura 4.18.

Sus entradas son:

- **negotiation:** Objeto con el contenido de la negociación de la cual se está mostrando el resumen, ver definición de *NegotiationEntity* en la Subsección 4.3.5.
- **user:** Objeto que contiene la información del usuario en sesión, ver definición de *User* en la Subsección 4.3.1.

4.6.15. OwnWorkExperience

Este componente de tipo tarjeta muestra de forma resumida el contenido de una experiencia laboral del usuario en sesión, así como la visibilidad de los campos. Además permite al usuario eliminar su experiencia laboral y navegar a la vista de edición de experiencias laborales. Este componente se muestra en la Figura 4.19.

Sus entradas son:

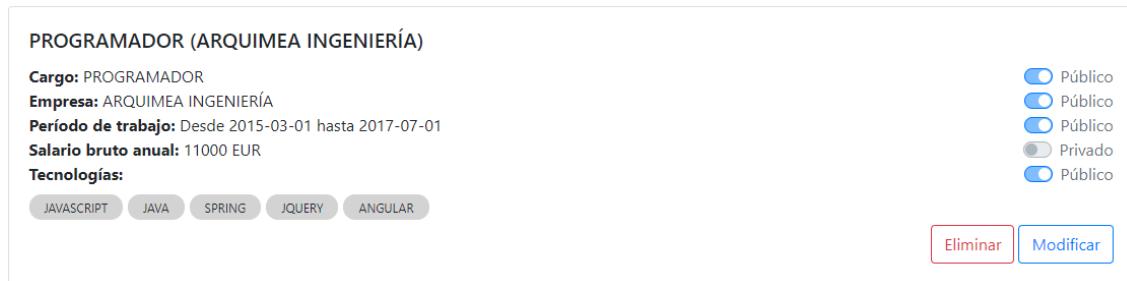


Figura 4.19: Ejemplo de componente web *OwnWorkExperience*.

- Hasta la actualidad
- Hasta...

Figura 4.20: Ejemplo de componente web *Radio*.

- **workExperience:** Objeto con el contenido de la experiencia laboral de la que se muestran los datos, ver definición de *WorkExperienceEntity* en la Subsección 4.3.3.
- **afterDelete:** Función de tipo callback que se ejecuta después del borrado de la experiencia laboral.
- **editable:** Si se establece a *true* permite eliminar y navegar a la vista de modificación de la experiencia laboral.

4.6.16. Radio

Este componente contiene una serie de botones de radio que permiten elegir entre varias opciones para establecer una variable. Este componente se muestra en la Figura 4.20.

Sus entradas son:

- **options:** Un objeto de tipo JSON que alimenta de opciones al componente. Cada clave representa el identificador de una posible opción, mientras que cada valor representa el texto que se mostrará para esta opción por pantalla.
- **selected:** Variable que contiene la opción seleccionada.
- **changeOption:** Función de tipo callback que se ejecuta cuando la opción del componente cambia.



Figura 4.21: Ejemplo de componente web *RequestButton*.

4.6.17. RequestButton

Este componente de tipo botón dispara peticiones HTTP cuando se hace click en él. Este componente se muestra en la Figura 4.21.

Sus entradas son:

- **text:** Texto del botón.
- **url:** URL a la que realizará las llamadas HTTP.
- **method:** Método HTTP, por defecto *GET*.
- **body:** Cuerpo de la solicitud HTTP, por defecto vacío.
- **headers:** Cabeceras de la solicitud HTTP, por defecto incluye la cabecera *Content-Type: application/json*.
- **styleClasses:** Clases de estilo adicionales que se le pueden añadir al botón.
- **onSuccess:** Función de tipo callback que se ejecuta cuando la petición ha terminado y permite manejar la respuesta.

4.6.18. Select

Este componente genera un elemento de tipo seleccionable que permite elegir entre varias opciones en un menú desplegable. Este componente se muestra en la Figura 4.22.

Sus entradas son:

- **values:** Un objeto de tipo JSON que alimenta de opciones al componente. Cada clave representa el identificador de una posible opción, mientras que cada valor representa el texto que se mostrará en el desplegable para esa opción.
- **onChange:** Función de tipo callback que se ejecuta cuando hay cambios en el seleccionable.

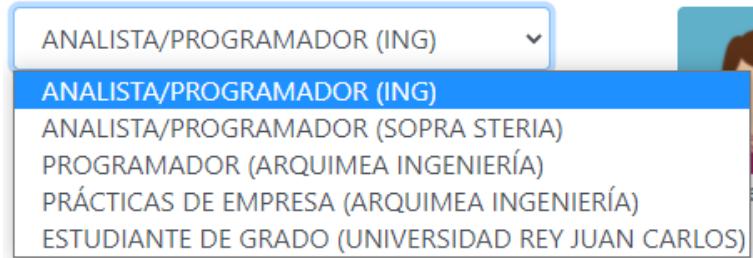


Figura 4.22: Ejemplo de componente web *Select*.

Figura 4.23: Ejemplo de componente web *WorkExperience*.

- **styleClasses:** Clases adicionales de estilos que se pueden añadir al componente.

4.6.19. WorkExperience

Este componente de tipo tarjeta muestra de forma resumida el contenido de una experiencia laboral resultado de una búsqueda. Este componente se muestra en la Figura 4.23.

Sus entradas son:

- **workExperience:** Objeto con el contenido de la experiencia laboral de la que se muestran los datos, ver definición de *WorkExperienceEntity* en la Subsección 4.3.3.
- **editable:** Si se establece a *true* habilita un botón que permite navegar a la vista de negociación.

4.6.20. WorkExperienceSummary

Este componente de tipo tarjeta muestra de forma resumida el contenido de una experiencia laboral vinculada a una negociación en la vista de detalles de negociación. El componente se muestra en la Figura 4.24.

The screenshot shows a user profile with a placeholder profile picture and the name 'vicengg'. To the right, the title 'ANALISTA/PROGRAMADOR (SOPRA STERIA)' is displayed. Below the title, there is a list of details: 'Cargo: ANALISTA/PROGRAMADOR', 'Empresa: SOPRA STERIA', 'Tecnologías: JAVA 11, SPRING BOOT', 'Salario: 100000€', and 'Período: 2018-02-01 - 2021-01-01'.

Figura 4.24: Ejemplo de componente web *WorkExperienceSummary*.

Sus entradas son:

- **workExperience:** Objeto con el contenido de la experiencia laboral de la que se muestran los datos, ver definición de *WorkExperienceEntity* en la Subsección 4.3.3.
- **visibilityRequest:** Variable que contiene la información de la visibilidad de la experiencia laboral de la que se muestran los datos, ver definición de *VisibilityRequest* en la Subsección 4.3.4.
- **setVisibilityRequest:** Función que establece el valor de la variable *visibilityRequest*.

4.7. Vistas

En esta sección se describen las vistas que componen la aplicación web. Para crear el sistema de vistas de la aplicación de una sola página, el enrutador de React asocia a cada vista una ruta de navegador concreta. Esta es la relación entre vistas y rutas dentro de la aplicación:

- Vista de *Mis experiencias*: Asociada a la ruta */my-work-experiences*.
- Vista de *Crear experiencia*: Asociada a la ruta */add-work-experience*.
- Vista de *Editar experiencia*: Asociada a la ruta */modify-work-experience/{workExperienceId}*, donde *workExperienceId* es el identificador único de la experiencia a editar.
- Vista de *Buscar experiencias*: Asociada a la ruta */search-work-experiences*.

- Vista de *Solicitud de información*: Asociada a la ruta `/create-information-request/{workExperienceId}`, donde `workExperienceId` es el identificador único de la experiencia sobre la que se quiere solicitar información.
- Vista de *Mis solicitudes*: Asociada a la ruta `/my-negotiations`.
- Vista de *Detalles de solicitud*: Asociada a la ruta `/negotiation-details/{negotiationId}`, donde `negotiationId` es el identificador único de la negociación de la que se consultan los detalles.

Como apoyo a la sección se incluye el diagrama de la Figura 4.25 que representa de forma esquemática la navegabilidad entre las distintas vistas.

4.7.1. Inicio de sesión

Esta vista es externa a la aplicación y es provista por GitHub. Su propósito es permitir que los usuarios inicien sesión con su cuenta de GitHub. Por lo tanto aquellos usuarios que traten de acceder a la *Vista de bienvenida* de la aplicación sin haber iniciado sesión previamente en GitHub serán redirigidos a esta vista. Una vez se introducen un usuario y clave de GitHub válido y se hace click en el botón *Sign in* que se muestra en la Figura 4.26 el usuario es conducido a la *Vista de bienvenida*.

4.7.2. Bienvenida

Es la primera vista de la aplicación, sirve como punto de entrada al resto de vistas. En ella el usuario ve su propio perfil de GitHub en sesión y puede leer una breve descripción de la herramienta. Una vez el usuario hace click en el botón *Empezar* que se muestra en la Figura 4.27 este es redirigido a la vista de *Mis experiencias*.

4.7.3. Mis experiencias

El propósito de esta vista es mostrar a cada usuario una lista con sus propias experiencias laborales, así como permitirle borrarlas o editarlas (redirigiéndole en este último caso a la vista de *Editar mi experiencia*). La primera vez que un usuario ingrese en la aplicación, al no poseer

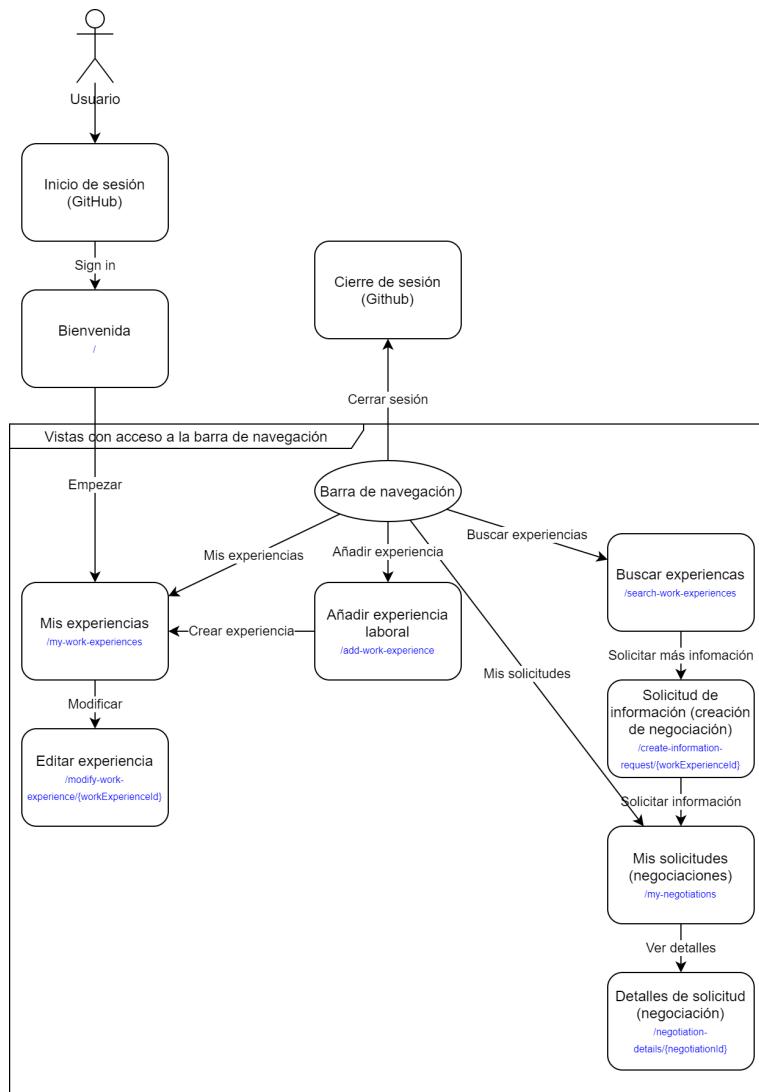


Figura 4.25: Diagrama de navegación entre las vistas de la aplicación.

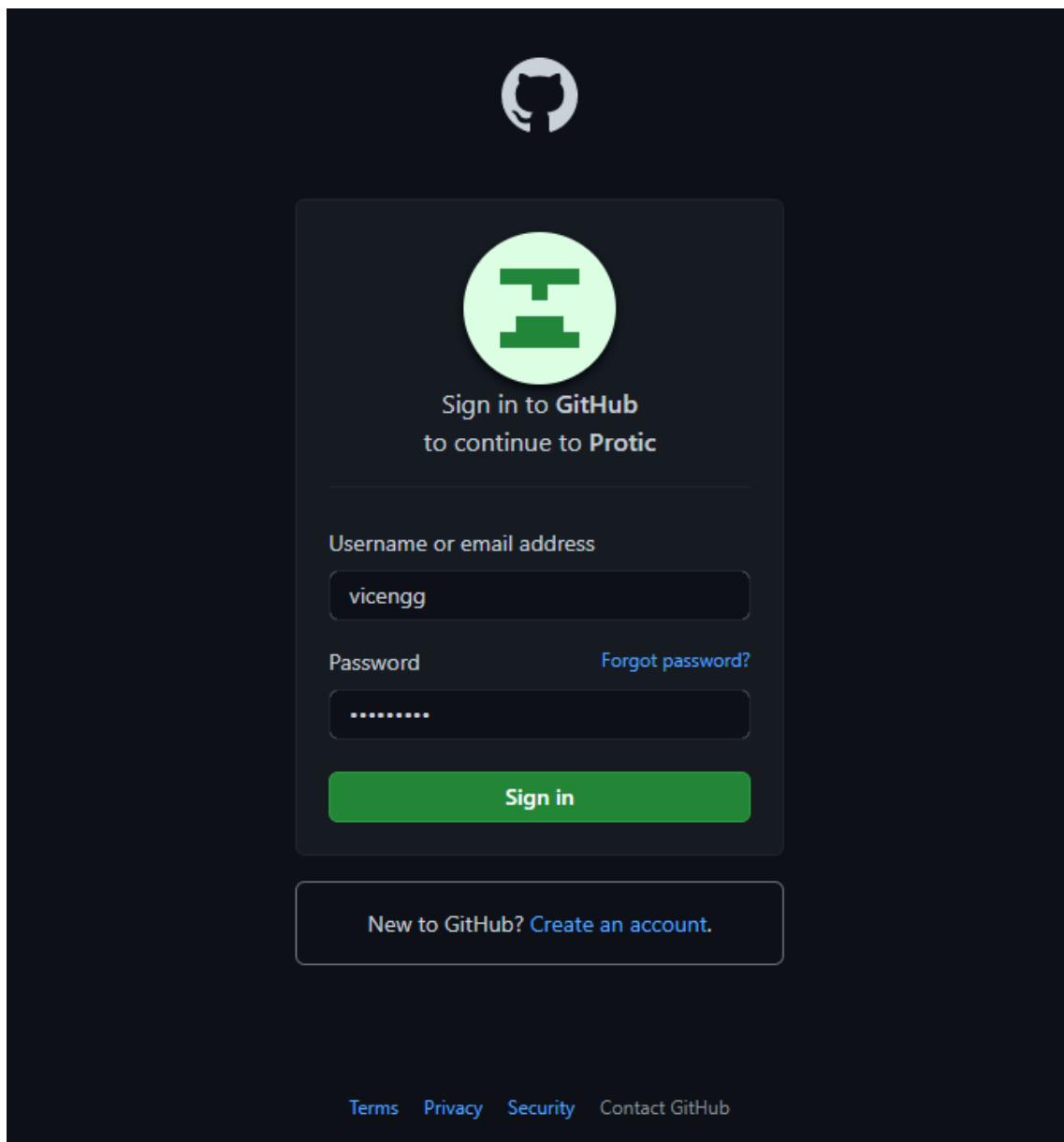


Figura 4.26: Vista de inicio de sesión.

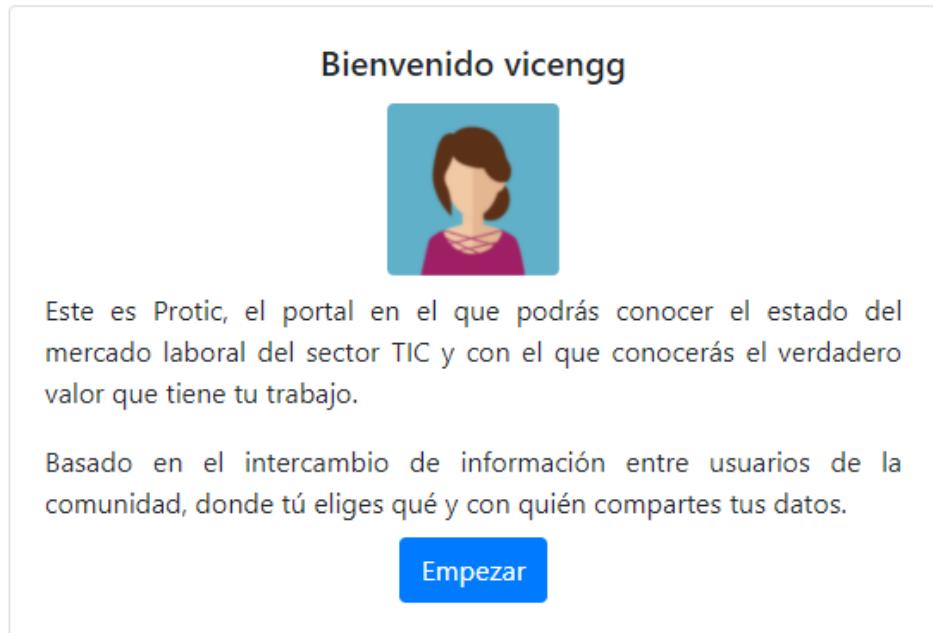


Figura 4.27: Vista de bienvenida.

ninguna experiencia añadida verá esta vista como la que se muestra en la Figura 4.29, desde donde podrá navegar a la vista de *Añadir experiencia*. Los usuarios que ya tengan experiencias laborales guardadas en la aplicación, verán esta vista como la que se muestra en la Figura 4.28. Desde aquí podrán eliminar cada una de sus experiencias haciendo click en el botón *Eliminar* y confirmando la operación en el modal emergente. También para cada experiencia, será redireccionado a la vista *Editar experiencia* haciendo click en el botón *Editar*.

Barra de navegación En adelante, todas las vistas de la aplicación tendrán acceso a la barra de navegación (ver Subsección 4.6.2), que permite navegar desde cualquier punto a:

- La vista de *Añadir experiencia*.
- La vista de *Mis experiencias*.
- La vista de *Buscar experiencia*.
- La vista de *Mis solicitudes*.
- La vista de *Cierre de sesión* de GitHub, invalidando previamente la sesión en la aplicación.

Mis experiencias laborales

ANALISTA/PROGRAMADOR (ING)

Cargo: ANALISTA/PROGRAMADOR
Empresa: ING
Período de trabajo: Desde 2021-01-01 hasta la actualidad
Salario bruto anual: 34000 EUR
Tecnologías:

- NODE
- MICROSOFT AZURE
- JAVA 11
- OPENSHIFT
- SPRING BOOT

Público
 Público
 Privado
 Privado
 Privado

[Eliminar](#) [Modificar](#)

ANALISTA/PROGRAMADOR (SOPRA STERIA)

Cargo: ANALISTA/PROGRAMADOR
Empresa: SOPRA STERIA
Período de trabajo: Desde 2018-02-01 hasta 2021-01-01
Salario bruto anual: 34000 EUR
Tecnologías:

- JAVA 11
- SPRING BOOT

Público
 Público
 Público
 Privado
 Público

[Eliminar](#) [Modificar](#)

PROGRAMADOR (ARQUIMEA INGENIERÍA)

Cargo: PROGRAMADOR
Empresa: ARQUIMEA INGENIERÍA
Período de trabajo: Desde 2015-03-01 hasta 2017-07-01
Salario bruto anual: 11000 EUR
Tecnologías:

- JAVASCRIPT
- JAVA
- SPRING
- JQUERY
- ANGULAR

Público
 Público
 Público
 Privado
 Público

Figura 4.28: Vista *Mis experiencias*.

Mis experiencias laborales

¡Aún no has añadido una experiencia laboral!

Para poder obtener información laboral de otros usuarios necesitas añadir tus propias experiencias laborales.

Puedes añadir tus experiencias laborales desde el panel "Añadir experiencia".

[Añadir experiencia](#)

Figura 4.29: Vista *Mis experiencias* vacía.

4.7.4. Añadir experiencia

Esta vista proporciona un formulario que permite al usuario crear una nueva experiencia laboral. Este formulario está compuesto por cinco campos como se muestra en la figura Figura 4.30:

- Puesto de trabajo: Nombre del puesto de trabajo de la experiencia laboral.
- Empresa: Nombre de la empresa de la experiencia laboral.
- Tecnologías: Listado de las tecnologías que se emplearon en la experiencia laboral.
- Salario: Salario bruto anual que se percibió en la experiencia laboral.
- Periodo laboral: Compuesto de una entrada de texto para la fecha de inicio, un botón de radio que permite indicar si la experiencia dura hasta el presente, y en caso negativo, otra entrada de texto para la fecha de fin.

Además cada campo tiene un interruptor que permite elegir si el campo tiene una visibilidad pública o privada para el resto de usuarios. Aquellos campos marcados como privados solo serán visibles para el propio usuario y para aquellos que hayan finalizado una negociación con él con el resultado de revelar dicho campo. Finalmente el formulario pregunta al usuario si desea vincular la experiencia que está creando con su perfil, en caso negativo esta experiencia aparecerá como anónima para el resto de usuarios. Una vez el usuario hace click en el botón *Crear experiencia laboral*, la experiencia será creada y el usuario redirigido a la vista *Mis experiencias*.

4.7.5. Editar experiencia

El propósito de esta vista es permitir a un usuario modificar datos de una de sus experiencias laborales. Esta vista proporciona un formulario igual que el de la vista *Añadir experiencia*, pero con los campos ya cumplimentados con la información de la experiencia laboral a editar para que el usuario los modifique. La vista está representada en la Figura 4.31. Una vez el usuario hace click en el botón *Guardar*, la experiencia será modificada y el usuario redirigido a la vista *Mis experiencias*.

Añadir experiencia laboral

Este panel te permite añadir una experiencia laboral. Recuerda que tú eliges quién puede ver tus datos: utiliza el **interruptor a la derecha** de cada campo para elegir si es visible para todo el mundo (**público**) o por el contrario solo es visible para ti y para aquellos usuarios a los que ofrezcas voluntariamente tu información (**privado**).

Protic te permitirá **solicitar** datos privados de otros usuarios y **ofrecer** los tuyos propios a fin de conseguir información laboral de otras personas, siempre por acuerdo mutuo y con el **nivel de anonimato** que elijas para tu experiencia laboral.

Información laboral

Profesión

<input type="text" value="Introduce tu profesión"/> <small>Haz click para obtener sugerencias de profesiones. Puedes introducir una profesión que no esté en la lista de sugerencias simplemente introduciendo su nombre.</small>	<input checked="" type="checkbox"/> Privado
---	---

Empresa

<input type="text" value="Introduzca una empresa"/> <small>Haz click para obtener sugerencias de empresas. Puedes introducir una empresa que no esté en la lista de sugerencias simplemente introduciendo su nombre.</small>	<input checked="" type="checkbox"/> Privado
--	---

Tecnologías

<input type="text" value="Introduzca un nombre de tecnología"/> <small>Haz click para obtener sugerencias de tecnologías. Puedes introducir múltiples tecnologías relacionadas con tu experiencia laboral. Para introducir una tecnología no sugerida, simplemente escribe su nombre y presiona la tecla "enter".</small>	<input checked="" type="checkbox"/> Privado
---	---

Salario

<input type="text" value="0"/> €	<input checked="" type="checkbox"/> Privado
---	---

Introduce tu salario bruto anual en Euros.

Periodo laboral

<input type="text" value="dd/mm/aaaa"/> ↻	<input checked="" type="radio"/> Hasta la actualidad	<input type="text" value="dd/mm/aaaa"/> <small>Introduce la fecha de fin (si procede).</small>	<input checked="" type="checkbox"/> Privado
--	--	--	---

Introduce la fecha de inicio.

Vincular experiencia con tu perfil de usuario

Vincular una experiencia con tu perfil permitirá al resto de usuarios conocer a quién pertenece esta experiencia, la verán asociada a tu nombre de usuario en el buscador de experiencias y en sus solicitudes de información.

No vincular esta experiencia a tu perfil te permitirá publicarla de forma anónima, de manera que nadie podrá saber a quién pertenece.

Vincular experiencia con perfil de usuario

No vinculada a tu perfil



[Cancelar](#)

[Crear experiencia laboral](#)

Figura 4.30: Vista Añadir experiencia.

Modificar experiencia laboral

Este panel te permite modificar una experiencia laboral. Recuerda que tú eliges quién puede ver tus datos: utiliza el **interruptor a la derecha** de cada campo para elegir si es visible para todo el mundo (**público**) o por el contrario solo es visible para ti y para aquellos usuarios a los que ofrezcas voluntariamente tu información (**privado**).

Protic te permitirá [solicitar](#) datos privados de otros usuarios y [ofrecer](#) los tuyos propios a fin de conseguir información laboral de otras personas, siempre por acuerdo mutuo y con el [nivel de anonimato](#) que elijas para tu experiencia laboral.

Información laboral

Profesión

ANALISTA/PROGRAMADOR

Público

Haz click para obtener sugerencias de profesiones. Puedes introducir una profesión que no esté en la lista de sugerencias simplemente introduciendo su nombre.

Empresa

ING

Público

Haz click para obtener sugerencias de empresas. Puedes introducir una empresa que no esté en la lista de sugerencias simplemente introduciendo su nombre.

Tecnologías

Introduzca un nombre de tecnología

Privado

Haz click para obtener sugerencias de tecnologías. Puedes introducir múltiples tecnologías relacionadas con tu experiencia laboral. Para introducir una tecnología no sugerida, simplemente escribe su nombre y presiona la tecla "enter".

NODE MICROSOFT AZURE JAVA 11 OPENSHIFT SPRING BOOT

Salario

100000

€

Privado

Introduce tu salario bruto anual en Euros.

Periodo laboral

01/01/2021



Hasta la actualidad

Hasta...

dd/mm/aaaa

Privado

Introduce la fecha de inicio.

Introduce la fecha de fin (si procede).

Vincular experiencia con tu perfil de usuario

Vincular una experiencia con tu perfil permitirá al resto de usuarios conocer a quién pertenece esta experiencia, la verán asociada a tu nombre de usuario en el buscador de experiencias y en sus solicitudes de información.

No vincular esta experiencia a tu perfil te permitirá publicarla de forma anónima, de manera que nadie podrá saber a quién pertenece.

Vincular experiencia con perfil de usuario

No vinculada a tu perfil



Anónimo



Figura 4.31: Vista *Editar experiencia*.

4.7.6. Buscar experiencias

Esta vista sirve al usuario para buscar experiencias laborales de otros usuarios. Para ello posee una serie de filtros en la parte superior como se muestra en la Figura 4.32:

- Puesto de trabajo: Filtra aquellas experiencias cuyo puesto de trabajo es un campo público y contiene la cadena de texto escrita en la entrada de texto.
- Empresa: Filtra aquellas experiencias cuya empresa es un campo público y contiene la cadena de texto escrita en la entrada de texto.
- Tecnologías: Filtra aquellas experiencias que contienen las tecnologías especificadas en el filtro.
- Salario mínimo: Filtra aquellas experiencias que tengan un salario superior al especificado.
- Salario máximo: Filtra aquellas experiencias que tengan un salario inferior al especificado.
- Fecha de entrada mínima: Filtra aquellas experiencias cuya fecha de entrada sea posterior a la especificada.
- Fecha de entrada máxima: Filtra aquellas experiencias cuya fecha de entrada sea anterior a la especificada.

Las experiencias que sean resultado de la búsqueda se presentaran en la vista como tarjetas (ver Subsección 4.6.19). Cada experiencia con campos privados presentará un botón *Solicitar más información* que permite navegar a la vista de *Solicitud de información*.

4.7.7. Solicitud de información

En esta vista, un usuario puede solicitar datos privados de una experiencia laboral de otro usuario, ofreciendo a la vez datos privados de una de sus experiencias laborales. Por ello la vista se divide en dos partes, como se muestra en la Figura 4.33:

Búsqueda experiencias laborales

Profesión Filtrar por profesión

Empresa Filtrar por empresa

Tecnologías Filtrar por tecnologías

Salario desde... € Introduzca un salario mínimo

Salario hasta... € Introduzca un salario máximo

dd/mm/aaaa Introduzca una fecha de entrada mínima

dd/mm/aaaa Introduzca una fecha de entrada hasta máxima

Experiencia laboral 

Cargo: Oculto 

Empresa: Oculto 

Período de trabajo: Desde 2020-01-01 hasta la actualidad

Salario bruto anual: 50000 EUR

Tecnologías:

DJANGO PYTHON



anónimo

[Solicitar más información](#)

ANALISTA/PROGRAMADOR (BABEL)

Cargo: ANALISTA/PROGRAMADOR

Empresa: BABEL

Período de trabajo: Desde 2018-02-01 hasta 2021-01-01

Salario bruto anual: Oculto 

Tecnologías:

JAVA 11 SPRING BOOT

[Solicitar más información](#)

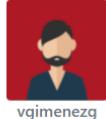
Figura 4.32: Vista *Buscar experiencias*.

Solicitud de información

Información ofrecida

<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;">ANALISTA/PROGRAMADOR (ING) </div> <p>Ofrece datos sobre una experiencia laboral propia para solicitar más información sobre la experiencia de otro usuario haciendo uso de los interruptores.</p> <p>Esta experiencia no está vinculada a tu perfil y por lo tanto será visualizada como de origen anónimo por el otro usuario.</p>	 vicengg
Cargo: ANALISTA/PROGRAMADOR	
Empresa: ING	
Tecnologías: NODE, MICROSOFT AZURE, JAVA 11, OPENSHIFT, SPRING BOOT	
Salario: 10000	
Período: 2021-01-01 - Actualidad	

Información solicitada

<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;">ANALISTA/PROGRAMADOR (BABEL)</div> <p>Solicita los datos de la experiencia laboral del usuario que deseas conocer haciendo uso de los interruptores. Puedes hacer más interesante la solicitud para el otro usuario ofreciendo tus propios datos.</p>	 vgimenezg
Cargo: ANALISTA/PROGRAMADOR	
Empresa: BABEL	
Tecnologías: JAVA 11, SPRING BOOT	
Salario:	
Período: 2018-02-01 - 2021-01-01	

Solicitar información

Figura 4.33: Vista *Solicitud de información*.

- Experiencia ofrecida: En la parte izquierda de la vista, el usuario solicitante puede elegir una de sus propias experiencias laborales en un seleccionador, y marcar aquellos campos que está dispuesto a revelar al usuario receptor de la solicitud.
- Experiencia solicitada: En la parte derecha de la vista, el usuario solicitante selecciona aquellos campos privados de la experiencia laboral del receptor que quiere que le sean revelados.

Una vez el usuario hace click en el botón *Solicitar información* será redirigido a la vista *Mis solicitudes*.

4.7.8. Mis solicitudes

Esta vista presenta la lista de negociaciones en las que el participa el usuario, ya sea porque ha creado solicitudes de información sobre experiencias de usuarios o bien porque otros usuarios han solicitado información de sus experiencias. Los usuarios que aun no participan en ninguna solicitud de información verán la vista como se muestra en la Figura 4.34. El resto de usuario verá una lista de tarjetas como la que se muestra en la figura Figura 4.35. Cada tarjeta

Mis solicitudes



Figura 4.34: Vista *Mis solicitudes* vacía.

presentada en la lista corresponde con una solicitud de información, en ella aparece el solicitante en el lado izquierdo de la tarjeta, el receptor de la solicitud en el lado derecho, el estado de la negociación en el centro de la tarjeta, y un resumen del último paso de la solicitud en la parte de abajo. Haciendo click en el botón *Ver detalles* de una tarjeta el usuario navegará a la vista de *Detalles de solicitud*.

4.7.9. Detalles de solicitud

En esta vista se presentan al usuario los detalles de una solicitud. La vista se divide entre alturas:

- Parte superior: A la izquierda se muestran el estado de la solicitud y el tiempo transcurrido desde que fue creada. A la derecha se muestran los botones con las posibles acciones que el usuario puede aplicar a la solicitud, como se muestra en la Figura 4.36, a distinguir:
 - Cancelar: Cancela la solicitud, cualquier usuario implicado en una solicitud pendiente tiene esta opción.
 - Aceptar: En la negociación por turnos, cuando es el turno de un usuario este puede aceptar la solicitud. Esto implica que se revelará la información ofrecida y solicitada en el último turno a ambos usuarios implicados.
 - Negociar: Cuando es el turno de un usuario, este puede negociar la solicitud para cambiar las condiciones impuestas por el otro usuario, es decir, el usuario puede

Mis solicitudes

The screenshot displays three cards representing user requests:

- Card 1:** Created by vicengg 10 minutes ago. Pending response from an anonymous user. Details: User anonymous requests data on technologies used and work period.
- Card 2:** Created by vicengg 10 minutes ago. Accepted by vgimenezg. Details: User vgimenezg shared data on technologies used and work period.
- Card 3:** Created by vicengg 3 hours ago. Cancelled by vgimenezg. Details: User vgimenezg shared salary data.

Figura 4.35: Vista *Mis solicitudes*.

modificar la visibilidad que se ofrece y solicita para los campos de las experiencias laborales asociadas a la solicitud. Hacer click en el botón de negociar hace que emerja un componente de tipo modal como el que se muestra en la Figura 4.37, similar a la vista de *Solicitud de información*. Cuando el usuario hace click en el botón *Enviar* del modal, se genera un nuevo paso en la solicitud, y el turno pasa al otro usuario implicado

- Parte media: A la izquierda se muestra una tarjeta con los detalles de la experiencia laboral ofrecida en la solicitud. A la derecha otra tarjeta con los detalles de la experiencia laboral solicitada.
- Parte inferior: A modo de conversación aparecen los distintos pasos que los usuarios implicados han ido dando en la negociación desde su creación hasta su estado actual, como se muestra en la Figura 4.38.

Detalles de solicitud

Creada: hace 32 minutos
Estado: Pendiente de tu respuesta

ANALISTA/PROGRAMADOR (ING)

anónimo

Puesto de trabajo: ANALISTA/PROGRAMADOR
Empresa: ING
Tecnologías: 🔒
Salario: 🔒
Período: 🔒

ANALISTA/PROGRAMADOR (BBVA)

vgimenezg

Puesto de trabajo: ANALISTA/PROGRAMADOR
Empresa: BBVA
Tecnologías: NODE, MICROSOFT AZURE, JAVA 11, OPENSHIFT, SPRING BOOT 🔒
Salario: 34000 EUR 🔒
Período: 2021-01-01 - Actualidad 🔒

anónimo creó la solicitud hace 32 minutos.

anónimo mostrará:

- Tecnologías
- Salario

vgimenezg mostrará:

- Salario

Cancelar **Aceptar** **Negociar**

Figura 4.36: Vista *Detalles de solicitud* durante la negociación.

Detalles

Creada: hace 59 minutos
Estado: Pendiente de respuesta

ANALISTA/PROGRAMADOR (ING)

anónimo

Puesto de trabajo: ANALISTA/PROGRAMADOR
Empresa: ING
Tecnologías: 🔒
Salario: 🔒
Período: 🔒

anónimo creó la solicitud hace 59 minutos.

anónimo mostrará:

- Tecnologías
- Salario

Negociar solicitud

ANALISTA/PROGRAMADOR (BBVA)

vgimenezg

Puesto de trabajo: ANALISTA/PROGRAMADOR
Empresa: BBVA
Tecnologías: NODE, MICROSOFT AZURE, JAVA 11, OPENSHIFT, SPRING BOOT 🔒
Salario: 34000 EUR 🔒
Período: 2021-01-01 - Actualidad 🔒

Cancelar **Enviar**

Figura 4.37: Vista *Detalles de solicitud* al modificar las condiciones de la solicitud.

Detalles de solicitud

Creada: hace 11 minutos

Estado: Aceptada

The screenshot shows a user interface for managing job requests. At the top, there are two separate boxes for job offers:

- ANALISTA/PROGRAMADOR (ING)**: Created by **vicengg**. Details: Cargo: ANALISTA/PROGRAMADOR, Empresa: ING, Tecnologías: NODE, MICROSOFT AZURE, JAVA 11, OPENSHIFT, SPRING BOOT, Salario: 34000 EUR, Período: 2021-01-01 - Actualidad.
- ANALISTA/PROGRAMADOR (BBVA)**: Created by **anónimo**. Details: Cargo: ANALISTA/PROGRAMADOR, Empresa: BBVA, Tecnologías: NODE, MICROSOFT AZURE, JAVA 11, OPENSHIFT, SPRING BOOT, Salario: 34000 EUR, Período: 2021-01-01 - Actualidad.

Below these, a central area shows the negotiation history:

- vicengg creó la solicitud hace 11 minutos.**
- vicengg** mostrará:
 - Tecnologías
 - Salario
 - Período laboral
- anónimo** mostrará:
 - Tecnologías
 - Salario
 - Período laboral
- anónimo negoció la solicitud hace 10 minutos.**
- vicengg** mostrará:
 - Tecnologías
 - Período laboral
- anónimo** mostrará:
 - Tecnologías
 - Período laboral
- vicengg aceptó la solicitud hace menos de 1 minuto.**

Figura 4.38: Vista *Detalles de solicitud* al concluir la negociación.

4.8. Casos de uso

En esta sección se procede a describir cada uno de los casos de uso que definen la funcionalidad de la aplicación. Estos casos de uso se muestran de forma esquemática en la Figura 4.39.

4.8.1. Actores

Para describir los casos de uso definiremos dos usuarios de ejemplo, llamados *Usuario 1* y *Usuario 2*. Ambos usuarios se registraron en GitHub previamente al ingreso a la aplicación. Para facilitar el seguimiento de los casos de uso se especifica que:

- *Usuario 1*: Es un usuario nuevo en la aplicación por lo que no posee experiencias laborales. En las figuras que acompañan a la explicación este usuario tendrá el nombre de usuario *vicengg* y un avatar con fondo azul. Interviene en todos los casos de uso.
- *Usuario 2*: Este usuario ya posee experiencias laborales guardadas en la aplicación. En las figuras que acompañan a la explicación este usuario tendrá el nombre de usuario *vigi-*

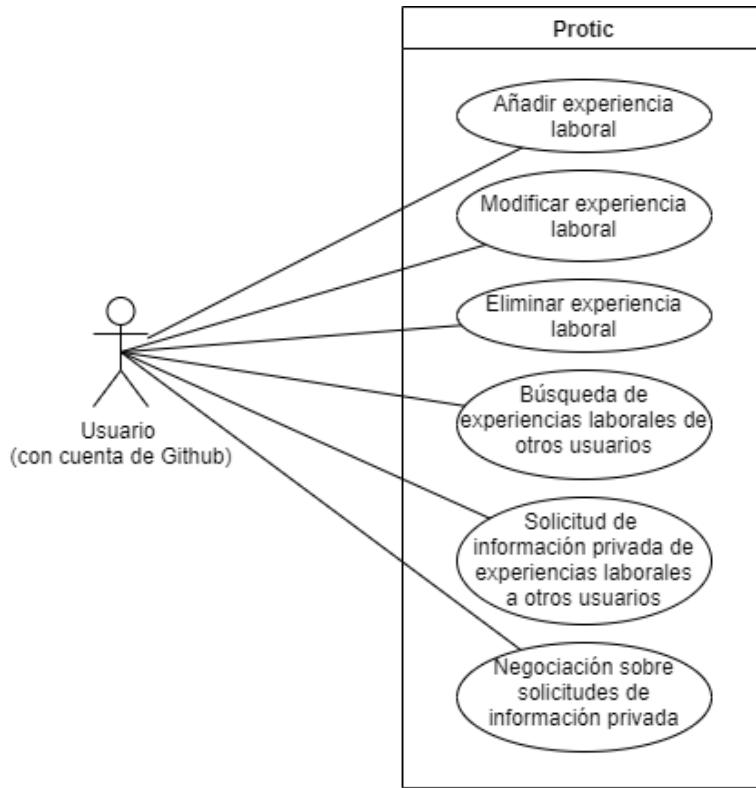


Figura 4.39: Casos de uso de la aplicación.

menezg y un avatar con fondo rojo.

4.8.2. Antecedentes: Ingreso en la aplicación

Durante la descripción de los casos de uso, siempre se asumirá que ambos usuarios han realizado los pasos de inicio de sesión que se describen a continuación y mantienen su sesión activa a la hora de interactuar en la aplicación:

1. El usuario fuera de sesión trata de acceder a la vista de bienvenida de la aplicación y es redirigido a la página de inicio de sesión de GitHub (Figura 4.40).
2. El usuario introduce sus credenciales de GitHub, hace click en el botón *Sign in* y es redirigido a la vista de bienvenida de la aplicación (Figura 4.41).
3. El usuario hace click en el botón *Empezar* y es redirigido a la vista de *Mis experiencias* donde puede acceder a la barra de navegación.

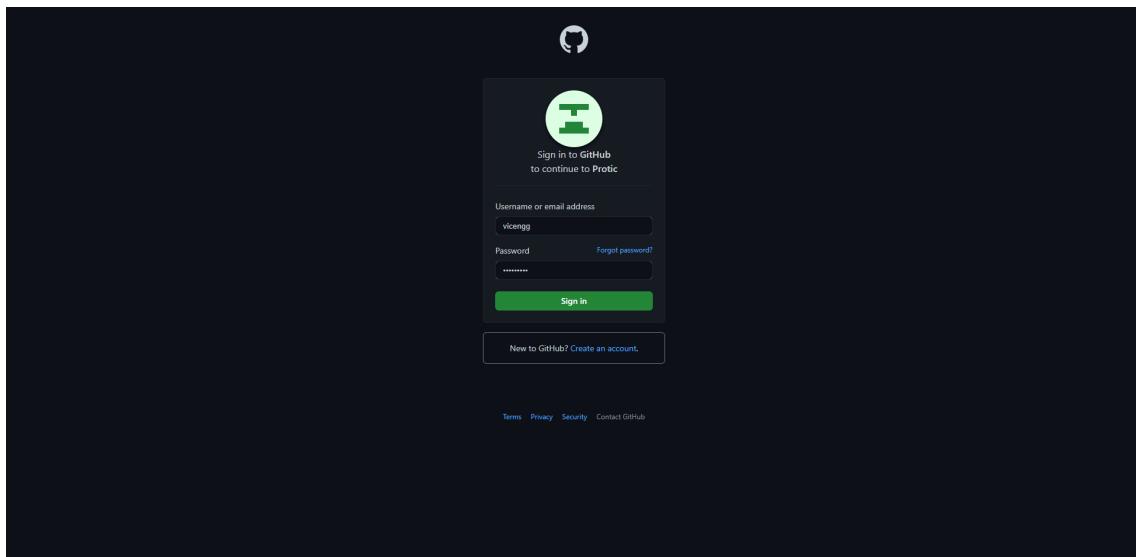


Figura 4.40: Página de inicio de sesión de GitHub.

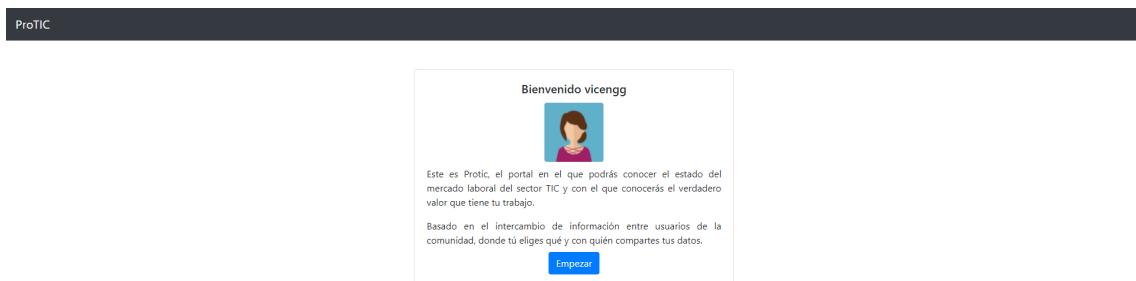


Figura 4.41: Vista de bienvenida de la aplicación.



Figura 4.42: El *Usuario 1* está en la vista *Mis experiencias*.

4.8.3. Añadir experiencia laboral

Para este escenario, el *Usuario 1* ha realizado los pasos que figuran en la Subsección 4.8.2 por lo que su sesión está iniciada y se encuentra en la vista *Mis experiencias* (Figura 4.42).

1. El *Usuario 1* hace click en el botón *Añadir experiencia* (en la barra de navegación o en el mensaje de la pantalla) y navega a la vista *Añadir experiencia*.
2. El *Usuario 1* rellena el formulario con la información de la experiencia laboral que quiere añadir (Figura 4.43).
3. El *Usuario 1* elige la visibilidad que desea para cada uno de los campos de su experiencia laboral, pública o privada.
4. El *Usuario 1* elige si quiere vincular la experiencia laboral a su perfil o prefiere que figure como anónima para el resto de usuarios.
5. El *Usuario 1* hace click en el botón *Crear experiencia laboral* y es redirigido de nuevo a la vista *Mis experiencias* donde puede ver su nueva experiencia laboral añadida (Figura 4.44).

Información laboral

Puesto de trabajo

Programador Público
 Privado

Haz click para obtener sugerencias de puestos de trabajo. Puedes introducir un puesto de trabajo que no esté en la lista de sugerencias simplemente introduciendo su nombre.

Empresa

Xampletic Privado
 Público

Haz click para obtener sugerencias de empresas. Puedes introducir una empresa que no esté en la lista de sugerencias simplemente introduciendo su nombre.

Tecnologías

Introduzca un nombre de tecnología
 Público

Haz click para obtener sugerencias de tecnologías. Puedes introducir múltiples tecnologías relacionadas con tu experiencia laboral. Para introducir una tecnología no sugerida, simplemente escribe su nombre y presiona la tecla "enter".

Java Javascript Spring Angular

Salario

18000 € Privado

Introduce tu salario bruto anual en Euros.

Periodo laboral

01/01/2021 Hasta la actualidad Público
 dd/mm/aaaa Hasta... Privado

Introduce la fecha de inicio. Introduce la fecha de fin (si procede).

Vincular experiencia con tu perfil de usuario

Vincular una experiencia con tu perfil permitirá al resto de usuarios conocer a quién pertenece esta experiencia, la verán asociada a tu nombre de usuario en el buscador de experiencias y en sus solicitudes de información.

No vincular esta experiencia a tu perfil te permitirá publicarla de forma anónima, de manera que nadie podrá saber a quién pertenece.

Vinculada a tu perfil
 Visible → 

Figura 4.43: El *Usuario 1* rellena el formulario con la información de la experiencia.

ProTIC Añadir experiencia Mis experiencias Buscar experiencias Mis solicitudes  vicengg ▾

Mis experiencias laborales

PROGRAMADOR (XAMPLETIC)

Puesto de trabajo: PROGRAMADOR
 Empresa: XAMPLETIC
 Período de trabajo: Desde 2021-01-01 hasta la actualidad
 Salario bruto anual: 18000 EUR
 Tecnologías:

Público
 Privado
 Público
 Privado
 Público

Figura 4.44: El *Usuario 1* ve su experiencia laboral añadida.

Figura 4.45: El *Usuario 1* modifica los campos de su experiencia.

4.8.4. Modificar experiencia laboral

Para este escenario, el *Usuario 1* ha creado al menos una experiencia laboral como se muestra en la Subsección 4.8.3 y se encuentra en la vista *Mis experiencias* (Figura 4.44) con experiencias en su lista.

1. El *Usuario 1* hace click en el botón *Modificar* y navega a la vista *Editar experiencia*.
2. El *Usuario 1* modifica en el formulario los campos de su experiencia que desea editar. (Figura 4.45).
3. El *Usuario 1* modifica las visibilidades que considera para sus campos.
4. El *Usuario 1* elige de nuevo si quiere vincular la experiencia laboral a su perfil o prefiere que figure como anónima para el resto de usuarios.
5. El *Usuario 1* hace click en el botón *Guardar* y es redirigido a la vista *Mis experiencias* donde puede ver su experiencia laboral modificada (Figura 4.46).

4.8.5. Eliminar experiencia laboral

Para este escenario, el *Usuario 1* ha creado al menos una experiencia laboral como se muestra en la Subsección 4.8.3 y se encuentra en la vista *Mis experiencias* (Figura 4.44) con expe-

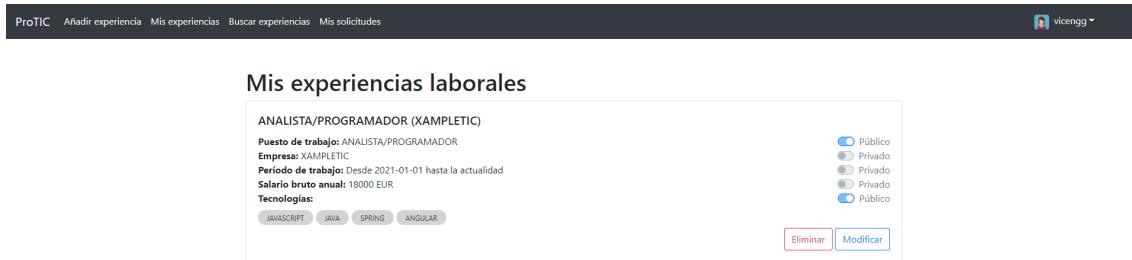


Figura 4.46: El *Usuario 1* ve su experiencia laboral modificada.

riencias en su lista.

1. El *Usuario 1* hace click en el botón *Eliminar* y un modal de confirmación aparece en pantalla.
2. El *Usuario 1* confirma el borrado haciendo click en el botón *Eliminar* del modal. (Figura 4.47).
3. El modal se cierra y el contenido de la vista se actualiza para mostrar que la experiencia laboral ha sido eliminada (Figura 4.48).

4.8.6. Búsqueda de experiencias laborales de otros usuarios

Para este escenario, el *Usuario 1* ha realizado los pasos que figuran en la Subsección 4.8.2 por lo que su sesión está iniciada y desde la barra de navegación accede a la vista *Buscar experiencias* (Figura 4.49).

1. El *Usuario 1* rellena los filtros en la parte superior de la pantalla para realizar una búsqueda de las experiencias que le interesan.
2. Cuando los filtros se rellenan la información de la página se actualiza automáticamente para mostrar los resultados (Figura 4.50).

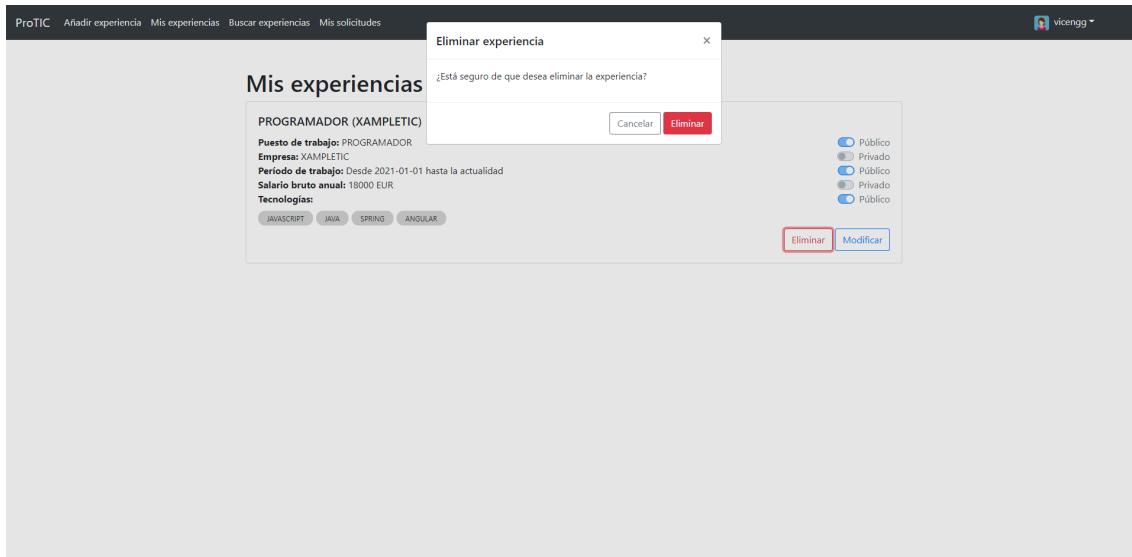


Figura 4.47: El *Usuario 1* confirma el borrado de su experiencia.



Figura 4.48: El *Usuario 1* ve que su experiencia laboral ha sido borrada.

Búsqueda experiencias laborales

Puesto de trabajo	Empresa	Tecnologías
Filtrar por puesto de trabajo	Filtrar por empresa	Filtrar por tecnologías
Salario desde... Introduzca un salario mínimo dd/mm/aaaa	Salario hasta... Introduzca un salario máximo dd/mm/aaaa	Introduzca una fecha de entrada hasta máxima
Introduzca una fecha de entrada mínima		
DEVOPS Puesto de trabajo: DEVOPS Empresa: Oculto ⓘ Periodo de trabajo: Desde 2021-05-01 hasta la actualidad Salario bruto anual: Oculto ⓘ Tecnologías: DOCKER MICROSOFT AZURE OPENSHIFT		
ANALISTA/PROGRAMADOR (BVA) Puesto de trabajo: ANALISTA/PROGRAMADOR Empresa: BVA Periodo de trabajo: Oculto ⓘ Salario bruto anual: Oculto ⓘ Tecnologías: Oculto ⓘ		
ANALISTA/PROGRAMADOR (SOPRA STERIA) Puesto de trabajo: ANALISTA/PROGRAMADOR Empresa: SOPRA STERIA Periodo de trabajo: Desde 2018-02-01 hasta 2021-01-01 Salario bruto anual: Oculto ⓘ Tecnologías: JAVA 11 SPRING BOOT		
ANALISTA/PROGRAMADOR (BABEL) Puesto de trabajo: ANALISTA/PROGRAMADOR Empresa: BABEL		

Figura 4.49: El *Usuario 1* está en la vista *Buscar experiencias*.

ProTIC [Añadir experiencia](#) [Mis experiencias](#) [Buscar experiencias](#) [Mis solicitudes](#) [vicensg](#)

Búsqueda experiencias laborales

Puesto de trabajo	Empresa	Tecnologías
Filtrar por puesto de trabajo	Filtrar por empresa	Filtrar por tecnologías
Salario desde... Introduzca un salario mínimo dd/mm/aaaa	Salario hasta... Introduzca un salario máximo dd/mm/aaaa	Introduzca una fecha de entrada hasta máxima
Introduzca una fecha de entrada mínima		
DEVOPS Puesto de trabajo: DEVOPS Empresa: Oculto ⓘ Periodo de trabajo: Desde 2021-05-01 hasta la actualidad Salario bruto anual: Oculto ⓘ Tecnologías: DOCKER MICROSOFT AZURE OPENSHIFT		

Figura 4.50: El *Usuario 1* filtra la búsqueda.

Figura 4.51: El *Usuario 1* está en la vista *Buscar experiencias*.

4.8.7. Solicitud de información privada de experiencias laborales a otros usuarios

Para este escenario, el *Usuario 1* ha creado al menos una experiencia laboral como se muestra en la Subsección 4.8.3 y se encuentra en la vista *Buscar experiencias* (Figura 4.51) con experiencias en su lista.

1. El *Usuario 1* selecciona una experiencia laboral de otro usuario (*Usuario 2*), hace click en el botón *Solicitar más información* y es redirigido a la vista de *Solicitud de información* (Figura 4.52).
2. En la parte superior de la tarjeta izquierda, el *Usuario 1* selecciona la experiencia laboral que desea ofrecer en el seleccionable.
3. En la parte inferior de la tarjeta izquierda, el *Usuario 1* elige los campos de su experiencia que está dispuesto a ofrecer al *Usuario 2*.
4. En la parte inferior de la tarjeta derecha, el *Usuario 1* elige los campos de la experiencia del *Usuario 2* que desea que le sean revelados.
5. El *Usuario 1* hace click en el botón *Solicitar información* y es redirigido a la vista *Mis solicitudes* (Figura 4.53) donde puede ver en la lista la entrada relativa a su nueva solicitud.

Solicitud de información

Información ofrecida

PROGRAMADOR (XAMPLETIC)	
Ofrece datos sobre una experiencia laboral propia para solicitar más información sobre la experiencia de otro usuario haciendo uso de los interruptores.	vicengg
Puesto de trabajo: PROGRAMADOR	<input checked="" type="checkbox"/> <input type="checkbox"/>
Empresa: XAMPLETIC	<input checked="" type="checkbox"/> <input type="checkbox"/>
Tecnologías: JAVASCRIPT, JAVA, SPRING, ANGULAR	<input checked="" type="checkbox"/> <input type="checkbox"/>
Salario: 18000 EUR	<input checked="" type="checkbox"/> <input type="checkbox"/>
Periodo: 2021-01-01 - Actualidad	<input checked="" type="checkbox"/> <input type="checkbox"/>

Información solicitada

DEVOPS	
Solicita los datos de la experiencia laboral del usuario que deseas conocer haciendo uso de los interruptores. Puedes hacer más interesante la solicitud para el otro usuario ofreciendo tus propios datos.	vgimenezg
Puesto de trabajo: DEVOPS	<input checked="" type="checkbox"/> <input type="checkbox"/>
Empresa:	<input checked="" type="checkbox"/> <input type="checkbox"/>
Tecnologías: DOCKER, MICROSOFT AZURE, OPENSHIFT	<input checked="" type="checkbox"/> <input type="checkbox"/>
Salario:	<input checked="" type="checkbox"/> <input type="checkbox"/>
Periodo: 2021-05-01 - Actualidad	<input checked="" type="checkbox"/> <input type="checkbox"/>

Solicitar información

Figura 4.52: El *Usuario 1* está en la vista *Solicitar información*.

Mis solicitudes

Creada por vicengg hace menos de 1 minuto	Esperando respuesta (de vgimenezg)	Recibida por vgimenezg
Has ofrecido a vgimenezg los siguientes datos: - Expresa	Has solicitado a vgimenezg los siguientes datos: - Salario	Ver detalles

Figura 4.53: El *Usuario 1* está en la vista *Mis solicitudes*.

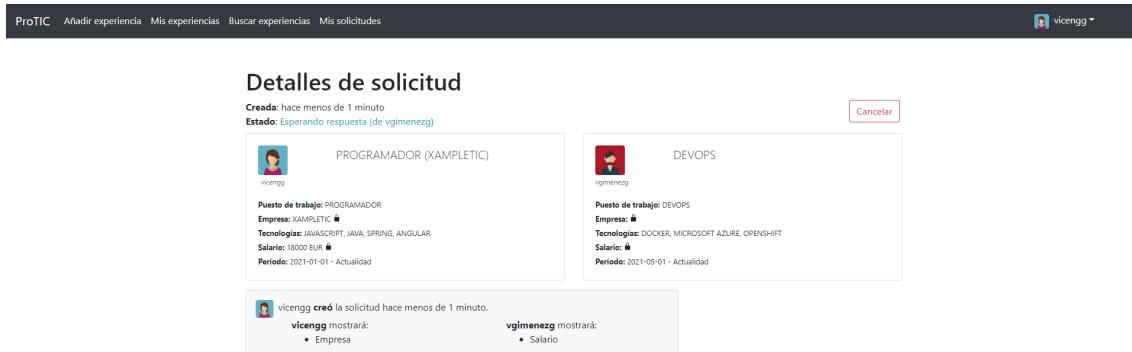


Figura 4.54: El *Usuario 1* está en la vista *Detalles de solicitud*.

6. El *Usuario 1* hace click en el botón *Ver detalles* y es redirigido a la vista *Detalles de solicitud* (Figura 4.54) donde comprueba los detalles de su solicitud recién creada.

4.8.8. Negociación sobre solicitudes de información privada

Para este escenario, el *Usuario 1* ha realizado los pasos descritos en la Subsección 4.8.8 por lo que tiene un solicitud de información creada sobre una experiencia laboral del *Usuario 2*.

1. El *Usuario 2* accede desde la barra de navegación a la vista *Mis solicitudes*, donde una solicitud aparece con el estado *Pendiente de tu respuestas* (Figura 4.55).
2. El *Usuario 2* hace click en el botón *Ver detalles* y es redirigido a la vista *Detalles de solicitud* (Figura 4.56).
3. El *Usuario 2* hace click en el botón *Negociar* y emerge una ventana modal que le permite modificar los términos de la solicitud (Figura 4.57).
4. El *Usuario 2* hace click en el botón *Enviar* para cambiar los términos de la negociación, y la solicitud pasa ahora al estado pendiente de respuesta por parte del *Usuario 1* (Figura 4.58). Esta nueva opción se refleja automáticamente en el histórico de la solicitud en la parte inferior de la vista. Los pasos 3 y 4 son opcionales, y se pueden repetir un número



Mis solicitudes

Creada por vicengg
hace 2 días

Pendiente de tu respuesta

Recibida por vgimenezg

El usuario **vicengg** te ofrece los siguientes datos:
- Salario

[Ver detalles](#)

Figura 4.55: El *Usuario 2* está en la vista *Mis solicitudes*.

Detalles de solicitud

Creada: hace 2 días
Estado: Pendiente de tu respuesta

[Cancelar](#) [Aceptar](#) [Negociar](#)

<p>PROGRAMADOR vicengg</p> <p>Puesto de trabajo: PROGRAMADOR Empresa: ● Tecnologías: JAVASCRIPT, JAVA, SPRING, ANGULAR Salario: ● Período: 2021-01-01 - Actualidad</p> <p>vgimenezg creó la solicitud hace 2 días. vicengg mostrará: • Empresa</p>	<p>DEVOPS (WARESOFT) vgimenezg</p> <p>Puesto de trabajo: DEVOPS Empresa: WARESOFT ● Tecnologías: DOCKER, MICROSOFT AZURE, OPENSHIFT Salario: 60000 EUR ● Período: 2021-05-01 - Actualidad</p> <p>vgimenezg mostrará: • Salario</p>
---	--

Figura 4.56: El *Usuario 2* está en la vista *Detalles de solicitud*.

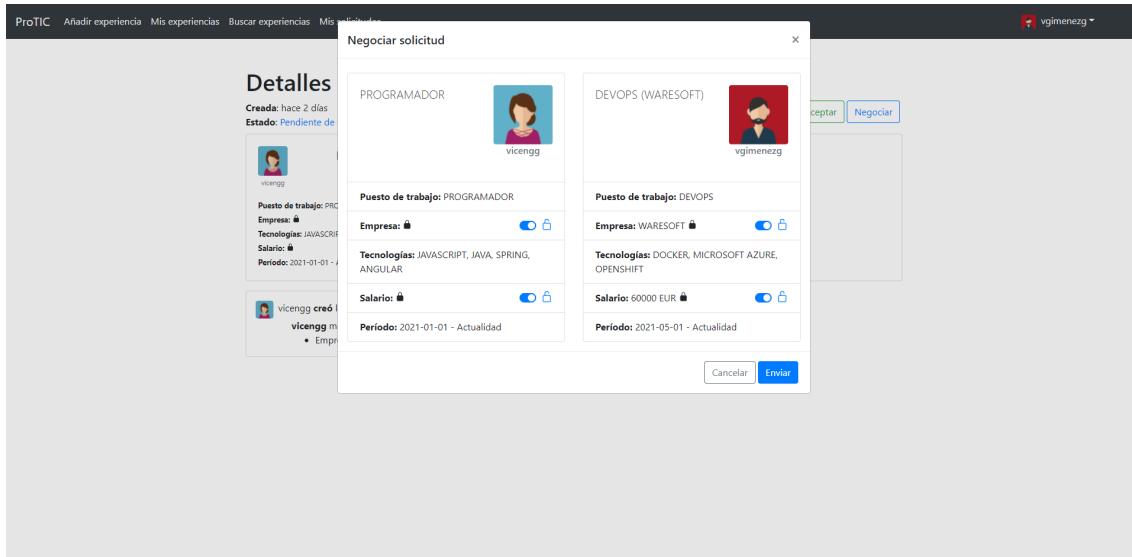


Figura 4.57: El *Usuario 2* negocia los términos de la solicitud en la ventana modal.

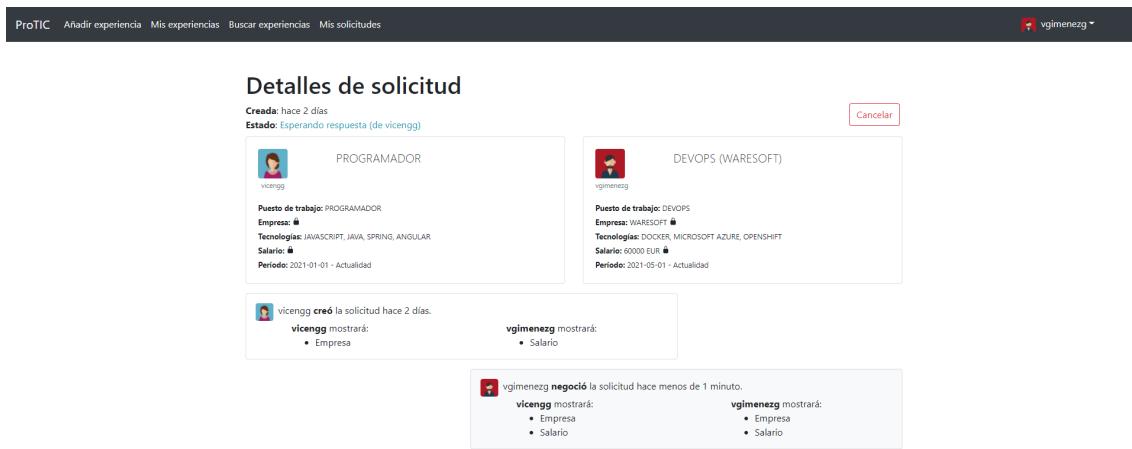


Figura 4.58: El estado de la solicitud cambia a pendiente de la respuesta del *Usuario 1*.

Detalles de solicitud

Creado: hace 2 días
Estado: Pendiente de tu respuesta

PROGRAMADOR (XAMPLETIC)
Puesto de trabajo: PROGRAMADOR
Empresa: XAMPLETIC
Tecnologías: JAVASCRIPT, JAVA, SPRING, ANGULAR
Salario: 18000 EUR
Periodo: 2021-01-01 - Actualidad

DEVOPS
Puesto de trabajo: DEVOPS
Empresa: WARESOFT
Tecnologías: DOCKER, MICROSOFT AZURE, OPENSHIFT
Salario: 60000 EUR
Periodo: 2021-05-01 - Actualidad

vgimenezg creó la solicitud hace 2 días.
vgimenezg mostrará:
• Empresa

vgimenezg mostrará:
• Salario

vgimenezg negoció la solicitud hace menos de 1 minuto.
vgimenezg mostrará:
• Empresa

vgimenezg mostrará:
• Salario

Figura 4.59: El *Usuario 1* visualiza el cambio en el histórico de la solicitud.

Detalles de solicitud

Creado: hace 2 días
Estado: Aceptada

PROGRAMADOR (XAMPLETIC)
Puesto de trabajo: PROGRAMADOR
Empresa: XAMPLETIC
Tecnologías: JAVASCRIPT, JAVA, SPRING, ANGULAR
Salario: 18000 EUR
Periodo: 2021-01-01 - Actualidad

DEVOPS (WARESOFT)
Puesto de trabajo: DEVOPS
Empresa: WARESOFT
Tecnologías: DOCKER, MICROSOFT AZURE, OPENSHIFT
Salario: 60000 EUR
Periodo: 2021-05-01 - Actualidad

vgimenezg creó la solicitud hace 2 días.
vgimenezg mostrará:
• Empresa

vgimenezg mostrará:
• Salario

vgimenezg negoció la solicitud hace 1 minuto.
vgimenezg mostrará:
• Empresa

vgimenezg mostrará:
• Salario

vicengg aceptó la solicitud hace menos de 1 minuto.

Figura 4.60: El *Usuario 1* acepta la solicitud y puede ver los campos de las experiencias desvelados.

indeterminado de veces alternando el turno entre el *Usuario 1* y el *Usuario 2*. Cada nueva negociación se añadirá al histórico.

5. El *Usuario 1* accede a la vista de detalles de la negociación, donde puede ver el nuevo paso en el histórico (Figura 4.59).
6. El *Usuario 1* hace click en el botón *Aceptar* para aceptar los nuevos términos de la solicitud, inmediatamente la vista se recarga mostrando los campos desvelados de la experiencia laboral del *Usuario 2* (Figura 4.60).
7. El *Usuario 2* del mismo modo puede comprobar los campos desvelados de la experiencia del *Usuario 1*.

En cualquier momento de este proceso previo a la aceptación ambos usuarios podrían cancelar la solicitud.

Capítulo 5

Pruebas y validación

Para la validación de la aplicación se siguieron dos estrategias distintas:

5.1. Pruebas manuales

Una vez realizado el desarrollo completo de la aplicación se aplicaron pruebas de usuario sobre la interfaz de usuario basadas en los casos de uso definidos en la Sección 4.8 y teniendo en cuenta los requisitos definidos en la Sección 2.3. Los defectos que se encontrar durante las pruebas manuales fueron corregidos hasta que la aplicación cumplió con todos los requisitos.

5.2. Pruebas automáticas de aceptación

Adicionalmente a las pruebas manuales, se implementaron unas pruebas automáticas utilizando el framework de pruebas Cucumber [26] con el propósito de detectar futuros errores derivados de la modificación de la lógica de negocio de la aplicación. Para ello se seleccionaron los requisitos que se consideraron críticos para el funcionamiento correcto de la aplicación, aquellos relacionados con la privacidad (requisitos 5, 6, 7, 8, 14 y 15 de la Sección 2.3). Se realizaron dos baterías de pruebas distintas:

- La primera comprueba la visibilidad de campos y la identidad de usuarios en experiencias laborales sobre las que no media ninguna solicitud de información.
- La segunda realiza las mismas comprobaciones en experiencias laborales sobre las que

median solicitudes de información que han sido aprobadas, se comprueba por lo tanto la visibilidad de los campos para usuarios involucrados en solicitudes aceptadas.

Las baterías de pruebas realizadas para esta sección se referencian abajo:

5.2.1. Pruebas sobre experiencias laborales sin solicitudes

Esta batería de pruebas comprueba el funcionamiento por defecto de la visibilidad de los campos de una experiencia laboral a la hora de mostrarse estos a los usuarios. Para los antecedentes de la prueba se crean dos usuarios distintos:

```
Feature: Work experience visibility by default
Check the work experiences visibility business rules by default.
```

```
Background:
Given a user with id "user_1"
And a user with id "user_2"
And the user "user_1" starts to fill the create work experience form
```

La primera prueba verifica que un usuario puede acceder a todos los campos y a su propia identidad de una experiencia laboral propia que contiene toda esta información marcada como pública.

```
Scenario: User 1 creates a work experience associated to his profile with all fields public.
When the user "user_1" fills the job title as "developer" with "public" visibility
And the user "user_1" fills the company as "Google" with "public" visibility
And the user "user_1" adds a technology "Java" with "public" visibility
And the user "user_1" adds a work period from "2016-08-16" to present with "public" visibility
And the user "user_1" adds a salary of 50000 "EUR" with "public" visibility
And the user "user_1" sends his work experience to create, associated to his profile, with reference ID "work_
When the user "user_1" gets his own work experiences
Then the work experience "work_experience_1" is present
And the work experience job title is present
And the work experience company is present
And the work experience technologies are present
And the work experience work period is present
And the work experience salary is present
And the work experience user is NOT anonymous
```

Esta prueba verifica que un usuario puede acceder a todos los campos y a su propia identidad de una experiencia laboral propia que es anónima y tiene marcados todos los campos como privados, ya que el mismo es el poseedor.

```
Scenario: User 1 creates a work experience anonymously with all fields private,
but he can see all data.
When the user "user_1" fills the job title as "developer" with "private" visibility
And the user "user_1" fills the company as "Google" with "private" visibility
And the user "user_1" adds a technology "Java" with "private" visibility
And the user "user_1" adds a work period from "2016-08-16" to present
    with "private" visibility
And the user "user_1" adds a salary of 50000 "EUR" with "private" visibility
And the user "user_1" sends his work experience to create, anonymously,
    with reference ID "work_experience_1"
When the user "user_1" gets his own work experiences
Then the work experience "work_experience_1" is present
And the work experience job title is present
And the work experience company is present
And the work experience technologies are present
And the work experience work period is present
```

```
And the work experience salary is present
And the work experience user is NOT anonymous
```

Esta prueba verifica que otro usuario puede acceder a todos los campos y a la identidad de una experiencia laboral ajena que contiene toda esta información marcada como pública.

```
Scenario: User 1 creates a work experience associated to his profile with all fields public,
then User 2 can see all data.
When the user "user_1" fills the job title as "developer" with "public" visibility
And the user "user_1" fills the company as "Google" with "public" visibility
And the user "user_1" adds a technology "Java" with "public" visibility
And the user "user_1" adds a work period from "2016-08-16" to present with "public" visibility
And the user "user_1" adds a salary of 50000 "EUR" with "public" visibility
And the user "user_1" sends his work experience to create, associated to his profile,
with reference ID "work_experience_1"
When the user "user_2" gets his own work experiences
Then the work experience "work_experience_1" is NOT present
When the user "user_2" gets work experiences of others
Then the work experience "work_experience_1" is present
And the work experience job title is present
And the work experience company is present
And the work experience technologies are present
And the work experience work period is present
And the work experience salary is present
And the work experience user is NOT anonymous
```

Esta prueba verifica que otro usuario NO puede acceder a ningún campo ni a la identidad de una experiencia laboral ajena que es anónima y tiene marcados todos los campos como privados.

```
Scenario: User 1 creates a work experience anonymously with all fields private,
then User 2 can't see any field nor the owner identity.
When the user "user_1" fills the job title as "developer" with "private" visibility
And the user "user_1" fills the company as "Google" with "private" visibility
And the user "user_1" adds a technology "Java" with "private" visibility
And the user "user_1" adds a work period from "2016-08-16" to present with "private" visibility
And the user "user_1" adds a salary of 50000 "EUR" with "private" visibility
And the user "user_1" sends his work experience to create, anonymously,
with reference ID "work_experience_1"
When the user "user_2" gets his own work experiences
Then the work experience "work_experience_1" is NOT present
When the user "user_2" gets work experiences of others
Then the work experience "work_experience_1" is present
And the work experience job title is NOT present
And the work experience company is NOT present
And the work experience technologies are NOT present
And the work experience work period is NOT present
And the work experience salary is NOT present
And the work experience user is anonymous
```

5.2.2. Pruebas sobre experiencias laborales con solicitudes aprobadas

Esta batería de pruebas comprueba el impacto sobre la visibilidad de los campos de las experiencias laborales para los usuarios cuando se acepta una solicitud de información. Para los antecedentes de la prueba se crean dos usuarios distintos, cada uno de ellos tiene una experiencia laboral con todos los campos privados y no vinculada a su perfil de usuario, es decir, anónima:

```
Feature: Work experience visibility with negotiations
Check the work experiences visibility business rules with negotiation.
```

Background:

```

Given a user with id "user_1"
And a user with id "user_2"
And the user "user_1" starts to fill the create work experience form
And the user "user_1" fills the job title as "developer" with "private" visibility
And the user "user_1" fills the company as "Google" with "private" visibility
And the user "user_1" adds a technology "Java" with "private" visibility
And the user "user_1" adds a work period from "2016-08-16" to present with "private" visibility
And the user "user_1" adds a salary of 50000 "EUR" with "private" visibility
And the user "user_1" sends his work experience to create, anonymously,
    with reference ID "work_experience_1"
And the user "user_2" starts to fill the create work experience form
And the user "user_2" fills the job title as "developer" with "private" visibility
And the user "user_2" fills the company as "Amazon" with "private" visibility
And the user "user_2" adds a technology "JavaScript" with "private" visibility
And the user "user_2" adds a work period from "2017-05-12" to present with "private" visibility
And the user "user_2" adds a salary of 60000 "EUR" with "private" visibility
And the user "user_2" sends his work experience to create, anonymously,
    with reference ID "work_experience_2"

```

En esta prueba se demuestra que el segundo usuario no ve ningún campo de la experiencia laboral del primero y que esta última se le muestra como anónima.

```

Scenario: Without negotiation, User 2 can't see anything about User 1's experience
When the user "user_2" gets work experiences of others
Then the work experience "work_experience_1" is present
And the work experience job title is NOT present
And the work experience company is NOT present
And the work experience technologies are NOT present
And the work experience work period is NOT present
And the work experience salary is NOT present
And the work experience user is anonymous

```

Para esta prueba el segundo usuario crea una solicitud de información solicitando visibilidad para todos los campos de la experiencia laboral del primer usuario y ofreciendo visibilidad a todos los campos de la experiencia laboral que él mismo ofrece. Cuando el primer usuario acepta la solicitud el resultado es que ambas experiencias laborales son totalmente visibles a ambos usuarios.

```

Scenario: User 2 creates a negotiation asking for visibility for all fields and offering
    all the fields of his work experience
When the user "user_2" creates a data request to unlock "work_experience_1" offering "work_experience_2"
And the user "user_2" adds a negotiation step
And asks for visibility for "job title, company, technologies, work period and salary"
And offers visibility for "job title, company, technologies, work period and salary"
And "user_2" sends the negotiation
And "user_1" accepts the negotiation
When the user "user_2" gets work experiences of others
Then the work experience "work_experience_1" is present
And the work experience job title is present
And the work experience company is present
And the work experience technologies are present
And the work experience work period is present
And the work experience salary is present
And the work experience user is anonymous
When the user "user_1" gets work experiences of others
Then the work experience "work_experience_2" is present
And the work experience job title is present
And the work experience company is present
And the work experience technologies are present
And the work experience work period is present
And the work experience salary is present
And the work experience user is anonymous

```

Para esta prueba el segundo usuario crea una solicitud de información solicitando visibilidad para algunos los campos de la experiencia laboral del primer usuario y ofreciendo visibilidad a algunos los campos de la experiencia laboral que él mismo ofrece. Cuando el primer usuario acepta la solicitud el resultado es cada usuario podrá ver sólo los campos de la experiencia laboral del otro que formaran parte del acuerdo.

```
Scenario: User 2 creates a negotiation asking for visibility for some fields of
the User 1's work experience and offering some fields of his work experience
When the user "user_2" creates a data request to unlock "work_experience_1" offering "work_experience_2"
And the user "user_2" adds a negotiation step
And asks for visibility for "job title, work period and salary"
And offers visibility for "company and technologies"
And "user_2" sends the negotiation
And "user_1" accepts the negotiation
When the user "user_2" gets work experiences of others
Then the work experience "work_experience_1" is present
And the work experience job title is present
And the work experience company is NOT present
And the work experience technologies are NOT present
And the work experience work period is present
And the work experience salary is present
And the work experience user is anonymous
When the user "user_1" gets work experiences of others
Then the work experience "work_experience_2" is present
And the work experience job title is NOT present
And the work experience company is present
And the work experience technologies are present
And the work experience work period is NOT present
And the work experience salary is NOT present
And the work experience user is anonymous
```


Capítulo 6

Conclusiones

6.1. Consecución de objetivos

Desde el punto de vista del objetivo general, la aplicación cumple con el propósito de permitir a estudiantes y trabajadores del sector TIC evaluar cuál es el rango salarial justo para una oferta de empleo mediante la utilización de esta para obtener información de otros usuarios con experiencias laborales similares a la oferta de empleo que se quiere estimar, ya que estas experiencias laborales pueden arrojar información sobre los principales parámetros que definen una oferta de empleo, es decir, puesto de trabajo, empresa, tecnologías empleadas, salario y período de actividad.

Desde el punto de vista de los objetivos específicos:

1. La aplicación permite a los usuarios añadir sus experiencias laborales a la plataforma, buscar experiencias laborales de otros usuarios y solicitar aquella información que no sea accesible en primer lugar.
2. La aplicación es accesible para todo usuario que posea una cuenta de GitHub, sin necesidad de registro adicional, algo común en estudiantes y empleados del sector TIC.
3. Para toda experiencia laboral, el usuario al que pertenece puede elegir en todo momento qué campos de ésta son visibles al resto de los usuarios y qué campos son privados y por lo tanto invisibles. Cuando un usuario solicita a otro usuario información privada, es este último el que tiene finalmente el control para revelarla o no hacerlo.

4. Los usuarios pueden operar con experiencias laborales propias que no vinculan a su perfil de usuario. De esta forma la experiencia laboral aparecerá como anónima tanto en la búsqueda de experiencias como en las solicitudes de información de forma que será imposible para el resto de usuarios averiguar a quién pertenece.
5. La plataforma proporciona un mecanismo de búsqueda basado en todos los filtros cualitativos y cuantitativos que componen una experiencia laboral de forma que resulta sencillo encontrar aquellas que son del interés de un usuario.
6. El mecanismo para obtener información pública de la plataforma es tan sencillo como realizar búsquedas con filtros. El mecanismo para obtener información privada de otros usuarios fomenta mediante la oferta de información privada propia que los usuarios se muestren receptivos a la hora de desvelar su información.

6.2. Lecciones aprendidas

Se ha desarrollado finalmente una aplicación que cumple con los objetivos y requisitos propuestos al inicio del desarrollo, es decir, permite a usuarios de las TIC intercambiar información laboral con control de su privacidad e identidad para así poder evaluar mejor el mercado laboral. Las lecciones aprendidas desde el punto de vista técnico son las siguientes se narran a continuación.

Desde el punto de vista del desarrollo de la aplicación web, la librería React demuestra ser una herramienta útil y potente que permite desarrollar componentes web fácilmente reutilizables (que de hecho en la mayoría de los casos se han reutilizado dentro de la propia aplicación de este proyecto) de forma muy rápida y orientada a proporcionar una buena experiencia de usuario. Uno de las principales desventajas que se han valorado y que no ha afectado a este proyecto es la manera que impone React de empotrar el HTML de los componentes en el código JavaScript lo que resulta positivo en cuanto a velocidad de desarrollo y encapsulamiento de los propios componentes, pero resultaría un problema a la hora de evolucionar en caso de necesidad los estilos CSS de los componentes de forma autónoma, sin tener que modificar el código fuente la aplicación, es decir, no permite trabajar sobre estilos de forma agnóstica al código JavaScript.

Atendiendo al uso de arquitectura hexagonal en el desarrollo del servidor, se establecen

como principales ventajas la facilidad que proporciona el diseño de capas a la hora de enfocarse en el diseño, desarrollo y pruebas de la lógica de negocio (modelo de dominio) sin tener que pensar en una primera fase en la implementación de cuestiones relacionadas con el transporte y persistencia de los datos (como las APIs REST o los conectores de las bases de datos), para poder realizar estos desarrollos a posteriori de la lógica de negocio y de forma autónoma de esta. Como principal desventaja se apunta a la verbosidad del código, que para poder cumplir con el diseño concéntrico en el que no existen dependencias de las capas internas hacia las más externas, obliga a duplicar en gran medida muchas estructuras de datos y a realizar mapeos de unas a las otras según la capa en la que se esté desarrollando, lo que genera gran cantidad de clases Java y código de mapeo. Esta desventaja podría paliarse utilizando lenguajes menos verbosos que Java, como Scala o Kotlin.

Para concluir, el uso de la plataforma GitHub como proveedor de autenticación y de información de usuario ha demostrado ser también muy ventajoso para el desarrollo del proyecto porque permite evitar implementar los mecanismos de registro, ingreso a la aplicación, mantenimiento de la sesión, estructuras para almacenamiento de los usuarios, etc., siendo provistas todas estas características por una plataforma persistente. El único punto negativo en este aspecto sería la necesidad de tener que desarrollar conectores ad-hoc si se quisiera integrar esta plataforma con otros proveedores con LinkedIn, ya que cada proveedor implementa el protocolo OAuth2 con distintas características que no permiten al conector utilizado para GitHub ser reutilizable.

6.3. Trabajos futuros

Para posibles fases posteriores a este Trabajo Fin de Grado se proponen los siguientes trabajos que podrían mejorar la aplicación:

- Agregar a la aplicación una tecnología de generación de contenedores, como Docker, que permita desplegar el servidor de aplicaciones, la aplicación web y la base de datos en un servidor accesible desde internet fácilmente.
- Realizar pruebas sobre la aplicación con usuarios reales.
- Añadir integración vía OAuth con LinkedIn adicionalmente a GitHub, de forma que los

usuarios con cuenta de LinkedIn también puedan acceder a la aplicación, lo que haría la funcionalidad más accesible a usuarios que no fueran del sector TIC.

- Hacer la aplicación totalmente responsive en dispositivos móviles. A pesar de que la aplicación está preparada para poder ser utilizada en dispositivos móviles sería posible mejorar la experiencia de usuario aplicando un diseño más amistoso para este tipo de dispositivos.
- Añadir validaciones en la interfaz de usuario de formato y obligatoriedad para la creación y modificación de experiencias. Se podría modificar los componentes que sirven de entrada de datos en los formularios para que estos mostraran errores al usuario caso de que falten campos obligatorios o no se hayan introducido en el formato correcto.
- Mostrar notificaciones cuando hay solicitudes de información nuevas en un punto de la aplicación que sea observable desde cualquier vista, como la barra de navegación.
- Añadir una descripción al modelo de experiencia laboral para poder intercambiar también información cualitativa.
- Implementar una funcionalidad de limpieza de filtros en la vista de búsqueda de experiencias laborales.

6.4. Código y estadísticas

El código fuente desarrollado para este Trabajo Fin de Grado se puede encontrar en el repositorio de GitHub del proyecto.¹.

El código del servidor de aplicaciones se compone principalmente de:

- 114 clases de Java con un total de 4555 líneas efectivas de código que contienen la lógica del servidor (test incluidos).
- 4 scripts SQL con un total de 558 líneas efectivas que contienen la estructura de la base de datos.
- 9 archivos XML con un total de 492 líneas efectivas que contienen

¹<https://github.com/vicengg/protic>

- 2 archivos de tipo *feature* con las definiciones Gherkin [5] (Cucumber) de las baterías de pruebas de aceptación. los mapeos de las operaciones de base de datos de MyBatis.

El código de la aplicación web se compone principalmente de:

- 40 ficheros JavaScript con un total de 2414 líneas efectivas de código que contienen la lógica de la aplicación web (componentes, hooks y helpers).
- 1 archivo CSS con 78 líneas para algunos de los estilos de la aplicación.
- 1 archivo HTML con 20 que carga los componentes de React y librerías.

Apéndice A

Mockups del diseño inicial

En este apéndice se muestran los *mockups* de los diseños antiguos de la aplicación, pertenecientes a la primera fase del proyecto que tuvo lugar en 2016.



Figura A.1: Vista inicial



Figura A.2: Vista inicio de sesión

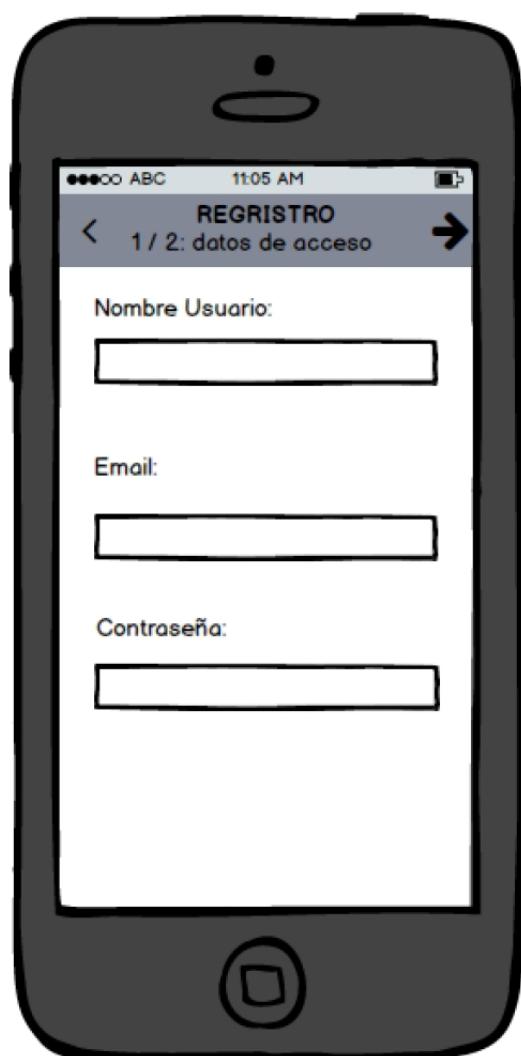


Figura A.3: Vista registro básico



Figura A.4: Vista registro completado



Figura A.5: Vista home

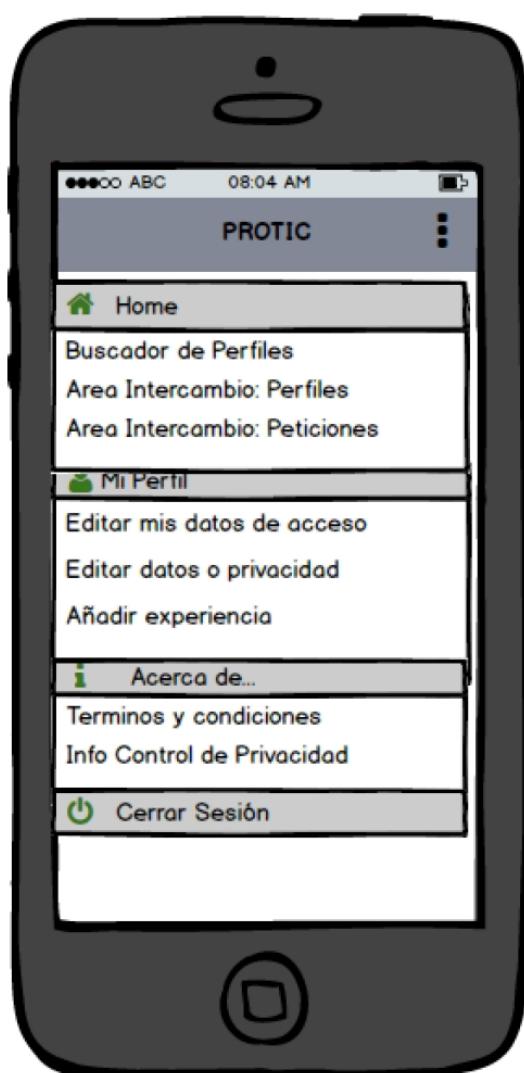


Figura A.6: Vista menú desplegable



Figura A.7: Vista búsqueda de perfiles



Figura A.8: Vista resultados de perfiles

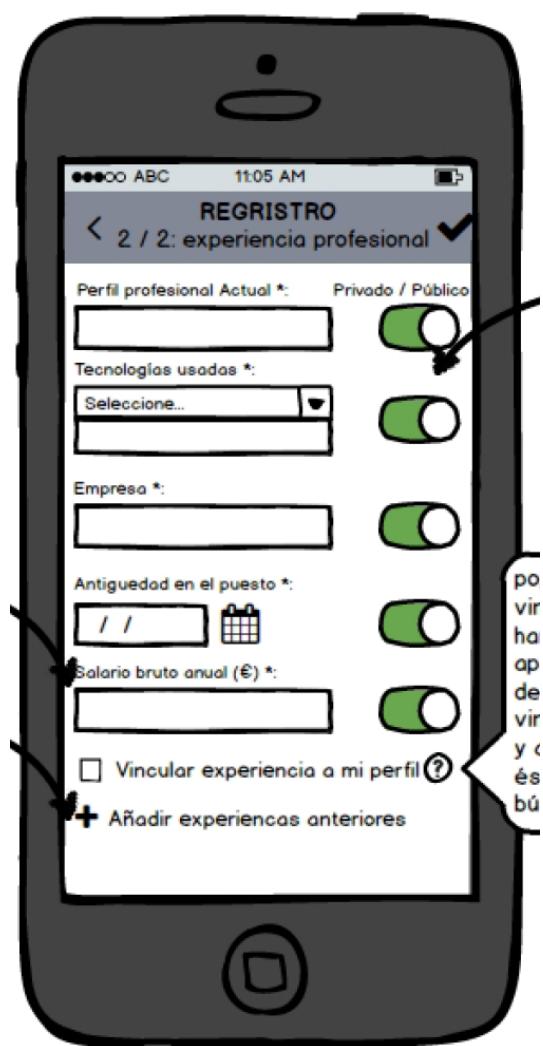


Figura A.9: Vista nueva experiencia laboral

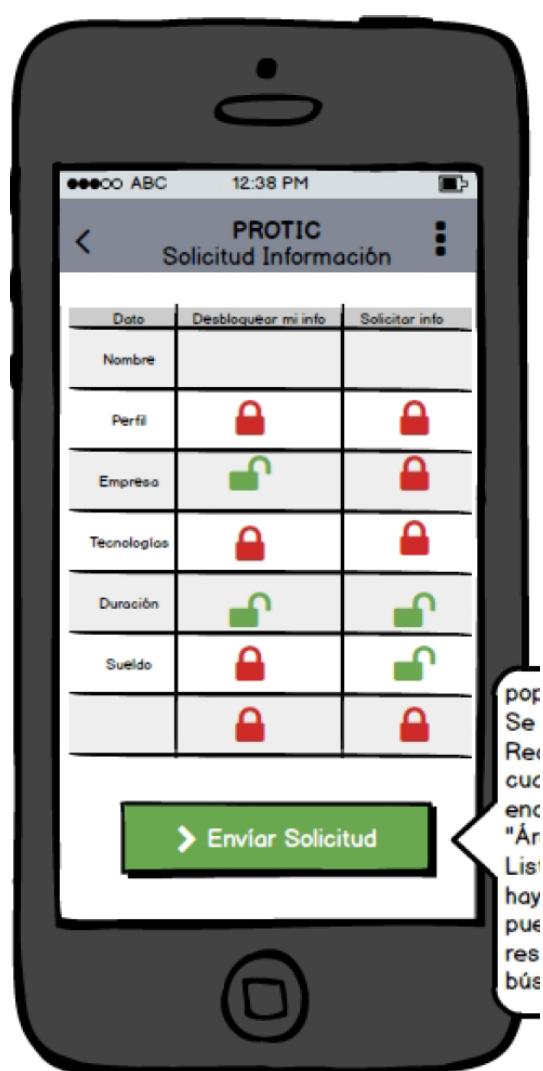


Figura A.10: Vista solicitud de negociación



Figura A.11: Vista solicitudes pendientes



Figura A.12: Vista informacion de perfiles laborales adquiridos



Figura A.13: Vista detalles de información laboral adquirida

Bibliografía

[1] CSS

<https://developer.mozilla.org/es/docs/Web/CSS>. Visitado por última vez el 14 de julio de 2021.

[2] DDD, Hexagonal, Onion, Clean, CQRS, ... How I put it all together

<https://herbertograca.com/2017/11/16/explicit-architecture-01-ddd-hex>
Visitado por última vez el 14 de julio de 2021.

[3] Domain Model

<https://martinfowler.com/eaaCatalog/domainModel.html>. Visitado por última vez el 14 de julio de 2021.

[4] Generalidades del protocolo HTTP

<https://developer.mozilla.org/es/docs/Web/HTTP/Overview>. Visitado por última vez el 14 de julio de 2021.

[5] Gherkin syntax

<https://cucumber.io/docs/gherkin/>. Visitado por última vez el 14 de julio de 2021.

[6] Herramienta de comparación de salario Glassdoor.com

<https://www.glassdoor.es/index.htm>. Visitado por última vez el 14 de julio de 2021.

[7] Herramienta de comparación de salario Payscale.com

<https://www.payscale.com/research/ES/Employer/Engineering-and-Science>. Visitado por última vez el 14 de julio de 2021.

[8] Herramienta de comparación de salario Salary.com

<https://www.salary.com/>. Visitado por última vez el 14 de julio de 2021.

[9] Herramienta de comparación de salario Salaryexpert.com

<https://www.salaryexpert.com/>. Visitado por última vez el 14 de julio de 2021.

[10] Hexagonal architecture

<https://alistair.cockburn.us/hexagonal-architecture/>. Visitado por última vez el 14 de julio de 2021.

[11] HTML: Lenguaje de etiquetas de hipertexto

<https://developer.mozilla.org/es/docs/Web/HTML>. Visitado por última vez el 14 de julio de 2021.

[12] Introduction to SQL

https://www.w3schools.com/sql/sql_intro.asp. Visitado por última vez el 14 de julio de 2021.

[13] Introduction to XML

https://www.w3schools.com/xml/xml_whatis.asp. Visitado por última vez el 14 de julio de 2021.

[14] Java EE at a Glance

<https://www.oracle.com/es/java/technologies/java-ee-glance.html>. Visitado por última vez el 14 de julio de 2021.

[15] JavaScript

<https://developer.mozilla.org/es/docs/Web/JavaScript>. Visitado por última vez el 14 de julio de 2021.

[16] Jvm: ¿Qué es y cómo funciona la Máquina Virtual Java?

<https://javadesdecero.es/fundamentos/como-funciona-maquina-virtual/>. Visitado por última vez el 14 de julio de 2021.

[17] Open Salaries: the Good, the Bad and the Awkward

<https://www.wsj.com/articles/open-salaries-the-good-the-bad-and-the-awkward>

Visitado por última vez el 14 de julio de 2021.

[18] Presentando JSX

<https://es.reactjs.org/docs/introducing-jsx.html>. Visitado por

última vez el 14 de julio de 2021.

[19] ¿Qué es el open source?

<https://www.redhat.com/es/topics/open-source/>

what-is-open-source. Visitado por última vez el 14 de julio de 2021.

[20] ¿Qué es Java?

https://www.java.com/es/about/whatis_java.jsp. Visitado por última

vez el 14 de julio de 2021.

[21] Qué es la programacion orientada a objetos.

<https://desarrolloweb.com/articulos/499.php>. Visitado por última vez

el 14 de julio de 2021.

[22] Qué es OAuth2

<https://openwebinars.net/blog/que-es-oauth2/>. Visitado por última

vez el 14 de julio de 2021.

[23] ¿Qué es una API de REST?

<https://www.redhat.com/es/topics/api/what-is-a-rest-api>. Visi-

tado por última vez el 14 de julio de 2021.

[24] Sitio web de Apache Maven

<https://maven.apache.org/>. Visitado por última vez el 14 de julio de 2021.

[25] Sitio web de Bootstrap

[https://getbootstrap.com/docs/4.0/getting-started/](https://getbootstrap.com/docs/4.0/getting-started/introduction/)

introduction/. Visitado por última vez el 14 de julio de 2021.

[26] Sitio web de Cucumber

<https://cucumber.io/>. Visitado por última vez el 14 de julio de 2021.

[27] Sitio web de Flyway

<https://flywaydb.org/>. Visitado por última vez el 14 de julio de 2021.

[28] Sitio web de Git

<https://git-scm.com/>. Visitado por última vez el 14 de julio de 2021.

[29] Sitio web de GitHub

<https://github.com/>. Visitado por última vez el 14 de julio de 2021.

[30] Sitio web de H2 database

<https://www.h2database.com/html/main.html>. Visitado por última vez el 14 de julio de 2021.

[31] Sitio web de IntelliJ IDEA

<https://www.jetbrains.com/es-es/idea/>. Visitado por última vez el 14 de julio de 2021.

[32] Sitio web de MyBatis

<https://mybatis.org/mybatis-3/es/>. Visitado por última vez el 14 de julio de 2021.

[33] Sitio web de NPM

<https://www.npmjs.com/>. Visitado por última vez el 14 de julio de 2021.

[34] Sitio web de Spring

<https://spring.io/>. Visitado por última vez el 14 de julio de 2021.

[35] Sitio web de Visual Studio Code

<https://code.visualstudio.com/>. Visitado por última vez el 14 de julio de 2021.

[36] Spring Social

<https://projects.spring.io/spring-social/>. Visitado por última vez el 14 de julio de 2021.

[37] Transparencia, por decreto

<https://mailchi.mp/bonillaware/transparencia-decreto?e=13a53236f5>. Visitado por última vez el 14 de julio de 2021.

[38] Understanding Domain Entities

<https://khalilstemmler.com/articles/typescript-domain-driven-design/entities/>. Visitado por última vez el 14 de julio de 2021.

[39] Value Objects

<https://medium.com/all-you-need-is-clean-code/value-objects-d4c24115fa69>. Visitado por última vez el 14 de julio de 2021.

[40] Web Components

https://developer.mozilla.org/es/docs/Web/Web_Components. Visitado por última vez el 14 de julio de 2021.

[41] What Is a Web Application? How It Works, Benefits and Examples

<https://www.indeed.com/career-advice/career-development/what-is-web-application>. Visitado por última vez el 14 de julio de 2021.

[42] What is an App Server?

<https://www.theserverside.com/news/1363671/What-is-an-App-Server>. Visitado por última vez el 14 de julio de 2021.

[43] When the Boss Says, 'Don't Tell Your Coworkers How Much You Get Paid'

<https://www.theatlantic.com/business/archive/2014/07/when-the-boss-says-dont-tell-your-coworkers-how-much-you-get-paid/374467/>. Visitado por última vez el 14 de julio de 2021.

[44] R. S. Rivero. Aplicación Móvil Android para el Intercambio de Información sobre Empleo entre Estudiantes con Controles de Privacidad, 2019.