



GRADO EN INGENIERÍA
MULTIMEDIA



VNIVERSITAT
DE VALÈNCIA

TRABAJO FIN DE GRADO

PLATAFORMA WEB PARA EL EMPAREJAMIENTO
DE PROFESIONALES Y PROYECTOS BASADA EN
COMPETENCIAS

AUTOR: JAVIER LEÓN SOLER

TUTOR: RAÚL PEÑA ORTIZ

JULIO 2025



VNIVERSITAT
DE VALÈNCIA



Escola Tècnica Superior
d'Enginyeria **ETSE-UV**

TRABAJO FIN DE GRADO

PLATAFORMA WEB PARA EL EMPAREJAMIENTO DE PROFESIONALES Y PROYECTOS BASADA EN COMPETENCIAS

AUTOR: JAVIER LEÓN SOLER

TUTOR: RAÚL PEÑA ORTIZ

Declaración de autoría:

Yo, Javier León Soler, declaro la autoría del [Trabajo Fin de Grado \(TFG\)](#) titulado “Plataforma web para el emparejamiento de profesionales y proyectos basada en competencias” y que el citado trabajo no infringe las leyes en vigor sobre propiedad intelectual. El material no original que figura en este trabajo ha sido atribuido a sus legítimos autores.

Valencia, 19 de abril de 2025

Fdo: Javier León Soler

Resumen:

Este es el resumen de [TFG](#). Debe ser corto (máximo media página) y cubrir los aspectos principales del [TFG](#).

Abstract:

This is the abstract of the [TFG](#). It must be short and cover the main aspects of the [TFG](#).

Resum:

Aquest és el resum del [TFG](#). Ha de ser curt (màxim mitja pàgina) i cobrir els aspectes principals del [TFG](#).

Agradecimientos:

En primer lugar quiero agradecer a todos aquellos que me han apoyado durante todos estos años.

En segundo lugar...

Índice general

1. Introducción	23
1.1. Introducción	23
1.2. Motivación	23
1.3. Objetivos	23
1.4. Organización de la memoria	23
2. Estado del arte	25
2.1. Análisis de aplicaciones similares	25
2.1.1. Plataformas web orientadas al empleo y la gestión del talento . . .	26
2.1.2. Aplicaciones con sistemas de emparejamiento basados en coincidencias	29
2.2. Evaluación de tecnologías	32
3. Requisitos, especificaciones, planificación, coste, riesgos y viabilidad	33
3.1. Requisitos	33
3.1.1. Requisitos funcionales	33
3.1.2. Requisitos no funcionales	33
3.2. Especificaciones	33
3.3. Planificación y estimación de costes	33
3.4. Riesgos	34
3.5. Viabilidad	34
4. Análisis	35
4.1. Diagrama de casos de uso	35
4.2. Diagrama de secuencia	35
4.3. Diagrama de clases de primer nivel	35
5. Diseño	37
5.1. Arquitectura de componentes	37
5.2. Despliegue del sistema	37
5.3. Modelo de datos	37

6. Implementación y pruebas	39
6.1. Implementación	39
6.2. Pruebas unitarias	39
6.3. Pruebas funcionales	39
6.4. Pruebas de rendimiento	39
6.5. Pruebas de usabilidad	39
6.6. Pruebas de seguridad	40
7. Conclusiones	41
7.1. Revisión de costes	41
7.2. Conclusiones	41
7.3. Trabajo futuro	41
8. Bibliografía	43
A. Glosarios	45
A.1. Acrónimos y abreviaciones	45
A.2. Términos	45
B. Ejemplos del lenguaje de marcado L^AT_EX	47
B.1. Algunos ejemplos básicos de uso de L ^A T _E X	47
B.2. Figuras y subfiguras	47
B.3. Tablas	48
B.4. Listados	51
B.5. Acrónimos y términos	52
C. Ejemplos de uso de B^IB_TE_X como gestor de bibliografía	55
C.1. Ejemplos de referencias formales bien valoradas	55
C.2. Ejemplos de referencias con valoración intermedia	55
C.3. Ejemplos de referencias con valoración baja a evitar en la medida de lo posible	55



Índice de figuras

2.1.	Interfaz de búsqueda de empleo en LinkedIn para “Programador Full Stack”.	27
2.2.	Ejemplo de selección de aptitudes en el perfil de usuario de LinkedIn.	27
2.3.	Ejemplo de oferta de trabajo publicada en InfoJobs.	28
2.4.	Interfaz principal de búsqueda de empleo en Yobalia, con filtros por ubicación, categoría y palabras clave.	29
2.5.	Selección opcional de intereses en Tinder durante la creación del perfil. . .	30
2.6.	Recomendaciones musicales personalizadas mediante mezclas diarias en Spotify.	31
2.7.	Selección de vídeos musicales adaptados a los gustos del usuario.	31
2.8.	Listado de ofertas laborales en Shapr ajustadas al perfil del usuario.	32
B.1.	Esta es una figura que latex decide donde colocar (floating) en el documento.	48
B.2.	Los cuatro diagramas en mosaico	48

Índice de tablas

B.1. Ejemplo de tabla 1	48
B.2. Web workload generators and grade in which main features are fulfilled . .	49
B.3. Ejemplo de tabla muy larga para una página	50

Índice de listados de código

B.1. Comandos bash para generar el fichero PDF	51
B.2. Comandos bash para generar el fichero PDF con arara	52
B.3. Ejemplo de listado en Python	52
B.4. Ejemplo de listado en Java	52
B.5. Ejemplo de listado en HTML 	52
B.6. Ejemplo de listado en CSS 	52

Capítulo 1

Introducción

1.1. Introducción

1.2. Motivación

1.3. Objetivos

1.4. Organización de la memoria

Capítulo 2

Estado del arte

En el presente capítulo se lleva a cabo un análisis del estado del arte con el fin de situar el proyecto en el contexto de soluciones existentes y tecnologías actuales. La plataforma web desarrollada en este TFG tiene como objetivo facilitar el emparejamiento entre profesionales y proyectos mediante un sistema basado en competencias jerarquizadas. Por ello, resulta esencial estudiar qué herramientas similares existen ya en el mercado y cómo se enfrentan a problemas de emparejamiento, recomendación o gestión de talento y proyectos.

Primero se presentarán aplicaciones con funcionalidades relacionadas, analizando sus características principales, puntos en común con esta propuesta y diferencias clave. Este estudio permitirá identificar tanto aspectos que se consideran imprescindibles en una plataforma de este tipo, como oportunidades de mejora o innovación.

Posteriormente se realizará un análisis crítico de las tecnologías más relevantes para el desarrollo del sistema, justificando la elección final en cada caso a partir de criterios como la escalabilidad, facilidad de desarrollo, mantenimiento o compatibilidad entre componentes.

Por último, se listarán aquellas herramientas de soporte utilizadas durante el desarrollo del TFG, como entornos de desarrollo, sistemas de control de versiones o editores de diagramas, las cuales han sido fundamentales para llevar a cabo el proyecto.

2.1. Análisis de aplicaciones similares

En esta sección se analizarán distintas aplicaciones existentes que presentan elementos comunes con la **plataforma web desarrollada en este trabajo**, cuyo objetivo es facilitar el **emparejamiento entre profesionales y proyectos** en función de sus competencias. Para ello, se comentarán **soluciones reales** que abordan problemas similares, ya sea desde la perspectiva de **plataformas orientadas al empleo y el talento**, o desde el enfoque de **sistemas de emparejamiento inteligente basados en afinidad**.

Dado que el proyecto combina ideas presentes en entornos profesionales como LinkedIn [1] y en algoritmos de emparejamiento como los utilizados por aplicaciones tipo Tinder [2], se ha optado por dividir este análisis en **dos bloques diferenciados**. En el primero se estudiarán **plataformas centradas en la gestión del talento y la búsqueda de empleo**, mientras que en el segundo se abordarán **sistemas cuyo núcleo es el emparejamiento basado en coincidencias**, con el objetivo de extraer ideas aplicables

al *Sistema de recomendaciones* que se desea implementar.

Este análisis permitirá no solo **identificar funcionalidades clave y enfoques existentes**, sino también detectar **carencias o posibles áreas de mejora**, con el fin de proponer una solución más **adaptada, automatizada** y centrada en la **coincidencia de competencias concretas**.

2.1.1. Plataformas web orientadas al empleo y la gestión del talento

Dado que la plataforma web a desarrollar tiene como objetivo conectar profesionales con proyectos en función de sus competencias, las primeras aplicaciones a analizar son aquellas centradas en la búsqueda de empleo, la exposición del perfil profesional y la gestión del talento.

En esta sección se analizarán concretamente las plataformas **LinkedIn** [1], **InfoJobs** [3] y **Yobalia** [4], destacando sus funcionalidades principales y su grado de similitud con la solución propuesta en este trabajo.

2.1.1.1. LinkedIn

En primer lugar, **LinkedIn** [1] es una plataforma web orientada al entorno profesional que permite a los usuarios crear un perfil con información detallada sobre su experiencia laboral, formación académica, certificaciones y competencias. Su objetivo principal es facilitar la conexión entre profesionales, empresas y oportunidades laborales, actuando como red social y como herramienta para la búsqueda de empleo.

Entre sus funcionalidades destaca un sistema de filtrado de ofertas que permite ajustar la búsqueda según múltiples parámetros, como ubicación, modalidad de trabajo (presencial o remoto), tipo de contrato o experiencia requerida. En la Figura 2.1 se muestra un ejemplo de búsqueda activa de empleo, donde se visualizan las ofertas disponibles y una vista previa de cada una de ellas. En particular, se ha realizado una búsqueda real orientada al perfil de **programador Full Stack** en la zona de **Valencia**, comprobando que la plataforma devuelve resultados actualizados y relevantes. Esta funcionalidad se basa en un sistema de filtrado tradicional implementado sobre formularios dinámicos en el *Frontend*, que construyen consultas específicas para recuperar ofertas desde su *Backend*.

Además, LinkedIn ofrece la posibilidad de añadir *habilidades* al perfil de usuario, las cuales pueden ser validadas por otros contactos, y que posteriormente pueden ser utilizadas por los reclutadores para filtrar candidatos en función de sus aptitudes. En la Figura 2.2 se ilustra esta sección del perfil. A pesar de ello, la plataforma no implementa un sistema de recomendaciones basado en coincidencias automáticas entre el perfil y las ofertas, sino que es el propio usuario quien debe llevar a cabo la búsqueda activa.

LinkedIn también dispone de herramientas de suscripción premium, que permiten acceder a estadísticas avanzadas sobre las postulaciones o enviar mensajes directos a reclutadores. Sin embargo, para el uso básico, la mayoría de funcionalidades son gratuitas.

En definitiva, LinkedIn representa un referente consolidado en la búsqueda de empleo y la gestión del talento, pero su funcionamiento está centrado en la exploración manual y el uso de filtros, sin un sistema automatizado de emparejamiento como el que se plantea en este trabajo.

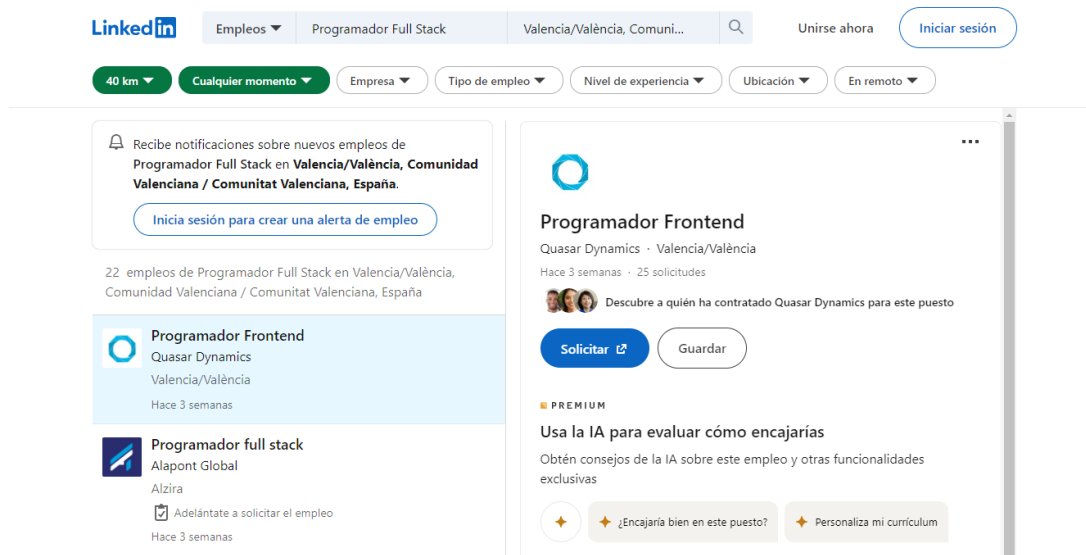


Figura 2.1: Interfaz de búsqueda de empleo en LinkedIn para “Programador Full Stack”.

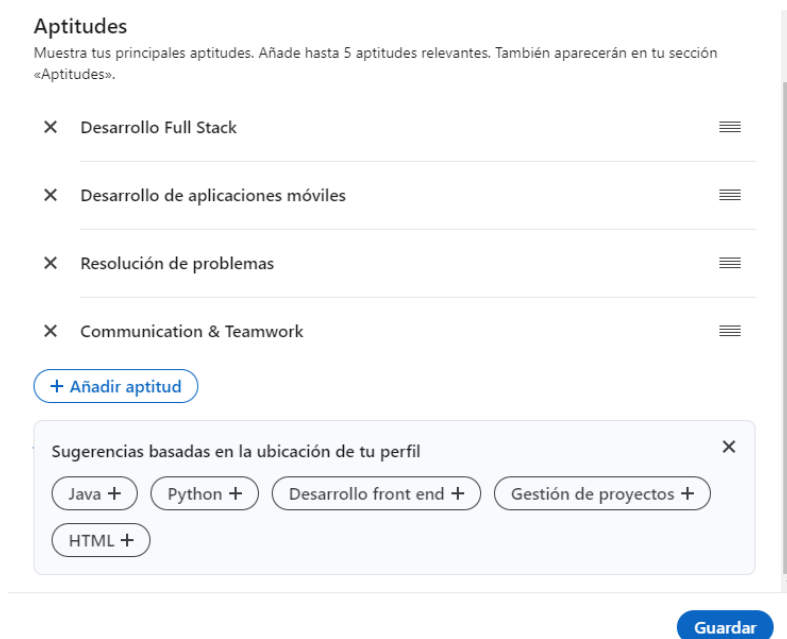


Figura 2.2: Ejemplo de selección de aptitudes en el perfil de usuario de LinkedIn.

2.1.1.2. InfoJobs

InfoJobs [3] es una plataforma generalista de búsqueda de empleo en España, orientada a la publicación y gestión de ofertas laborales de múltiples sectores. A diferencia de **LinkedIn**, su funcionamiento se centra en la inscripción directa del usuario a las ofertas, sin red de contactos ni validación de aptitudes.

Este sistema de búsqueda utiliza formularios predefinidos en el frontend para generar filtros que se aplican sobre la base de datos mediante consultas directas al backend, priorizando coincidencias exactas entre los criterios seleccionados y los campos del perfil del usuario. Sin embargo, no emplea ningún tipo de lógica avanzada de emparejamiento automático o inferencia basada en competencias.

En la Figura 2.3 se muestra un ejemplo de oferta publicada para un puesto de programador web, donde se observan los detalles del puesto, los requisitos y las opciones de inscripción.

Frente a este enfoque, la plataforma propuesta en este TFG plantea un sistema de recomendaciones que busca automatizar el emparejamiento, priorizando aquellas coincidencias más relevantes entre las competencias del profesional y las necesidades del proyecto.

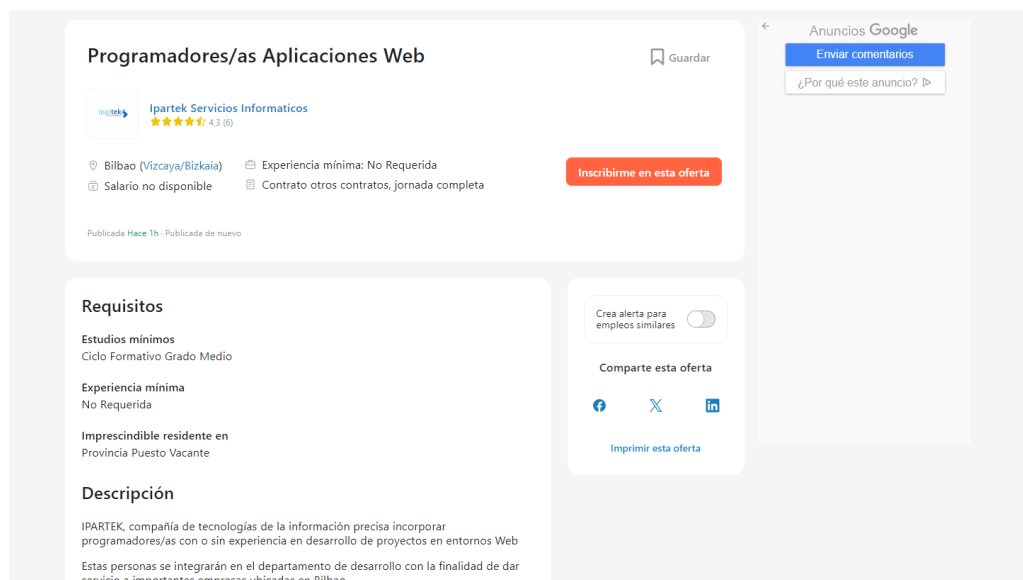


Figura 2.3: Ejemplo de oferta de trabajo publicada en InfoJobs.

2.1.1.3. Yobalia

Por último, **Yobalia** [4] es una plataforma web de búsqueda de empleo especializada en el sector de eventos, promociones y acciones publicitarias. Está orientada principalmente a perfiles como azafatas, promotores, animadores o personal para campañas de marketing en punto de venta.

Desde su página principal queda patente esta orientación, posicionándose como un portal de referencia para trabajos puntuales y de corta duración. En la Figura 2.4 se muestra su interfaz de búsqueda, donde es posible filtrar las ofertas disponibles mediante criterios básicos como la ubicación, la categoría profesional o palabras clave introducidas por el usuario. Este último elemento guarda cierta relación con el sistema basado en competencias que plantea la solución de este TFG, si bien su funcionamiento se basa en una coincidencia textual directa sin ningún tipo de análisis semántico o ponderación.

Aunque permite registrar el currículum y aplicar de forma rápida a las ofertas, la plataforma carece de validación de habilidades, conexión entre perfiles o mecanismos automatizados de emparejamiento. Todo el proceso de búsqueda y postulación depende de la acción manual del usuario.

En este sentido, la plataforma propuesta en este trabajo no solo abarca perfiles técnicos, sino que también está concebida para facilitar el acceso a trabajos eventuales como los ofrecidos en Yobalia, mejorando la eficiencia del proceso mediante un sistema de recomendaciones que prioriza las coincidencias entre requisitos y competencias de forma más automatizada.



Figura 2.4: Interfaz principal de búsqueda de empleo en Yobalia, con filtros por ubicación, categoría y palabras clave.

2.1.1.4. Conclusiones

Tras analizar las plataformas presentadas, podría cuestionarse si hubiera sido más eficiente utilizar directamente alguna de ellas para gestionar el emparejamiento entre profesionales y proyectos. Sin embargo, existen motivos claros para optar por una **plataforma desarrollada a medida**.

LinkedIn, aunque potente, es una red generalista donde el emparejamiento entre usuarios y ofertas no se basa en un **sistema automático guiado por competencias**, sino en la búsqueda activa y manual. **InfoJobs**, por su parte, se basa en filtros simples definidos por las ofertas, sin aplicar lógica de emparejamiento basada en competencias reales. Finalmente, **Yobalia** está centrada en trabajos puntuales y carece de un **sistema de recomendación avanzado**.

Por tanto, desarrollar una solución propia permite construir un modelo **centrado en competencias**, adaptable a distintos tipos de perfiles y proyectos, y con capacidad de evolución. Esta decisión garantiza una mayor **flexibilidad** y un **control total** sobre la funcionalidad y el enfoque del sistema, ajustándose plenamente a los objetivos de este TFG.

2.1.2. Aplicaciones con sistemas de emparejamiento basados en coincidencias

Una vez analizadas las plataformas orientadas al empleo y la gestión del talento, es interesante estudiar ciertas aplicaciones que, aunque no estén diseñadas con un fin profesional, hacen uso de sistemas de emparejamiento o recomendación basados en la coincidencia entre perfiles, intereses o preferencias. Estas soluciones ofrecen ideas relevantes que pueden ser adaptadas a un contexto donde las competencias técnicas son el eje central del emparejamiento.

En este apartado se analizarán tres casos representativos. En primer lugar, **Tinder** [2], como referente en emparejamiento entre personas a través de la coincidencia de interés mutuo. A continuación, **Spotify** [5], cuyo sistema de recomendaciones personalizadas brinda conceptos aplicables (sin entrar en la complejidad de sus algoritmos internos). Por

último, **Shapr** [6], que traslada la lógica de afinidad al entorno profesional, sugiriendo perfiles compatibles según objetivos e intereses compartidos.

El objetivo es extraer elementos útiles como la personalización, la afinidad o la lógica de coincidencia, con el fin de incorporarlos a un sistema más estructurado y adaptado al emparejamiento entre profesionales y proyectos según sus competencias.

2.1.2.1. Tinder

Tinder [2] es una aplicación centrada en el emparejamiento entre personas mediante la coincidencia de intereses. El sistema sugiere perfiles según criterios como edad, género y ubicación, y cada usuario decide si le interesa o no. Cuando ambas personas muestran interés, se genera un *Match* que permite iniciar una conversación.

Parte de la lógica que sigue Tinder está basada en los gustos personales, que pueden indicarse al crear el perfil, como se muestra en la Figura 2.5. Sin embargo, esta información es completamente opcional y no condiciona el funcionamiento del sistema, que se basa principalmente en la acción voluntaria del usuario, recogiendo principalmente otros parámetros obligatorios como la ubicación, edad, tipo de relación, etc.

La plataforma propuesta en este trabajo recoge esa idea de emparejar por afinidad, pero la lleva a un terreno más técnico. En lugar de basarse únicamente en preferencias generales o gustos personales, los profesionales deben definir competencias concretas mediante palabras clave, además de otros datos como la ubicación o el tipo de proyecto deseado. A diferencia de Tinder, donde el emparejamiento depende de la acción mutua de los usuarios, aquí se realiza de forma automática en función del grado de coincidencia entre los perfiles y los requisitos definidos, permitiendo una búsqueda más eficiente y adaptada a las necesidades de cada usuario.

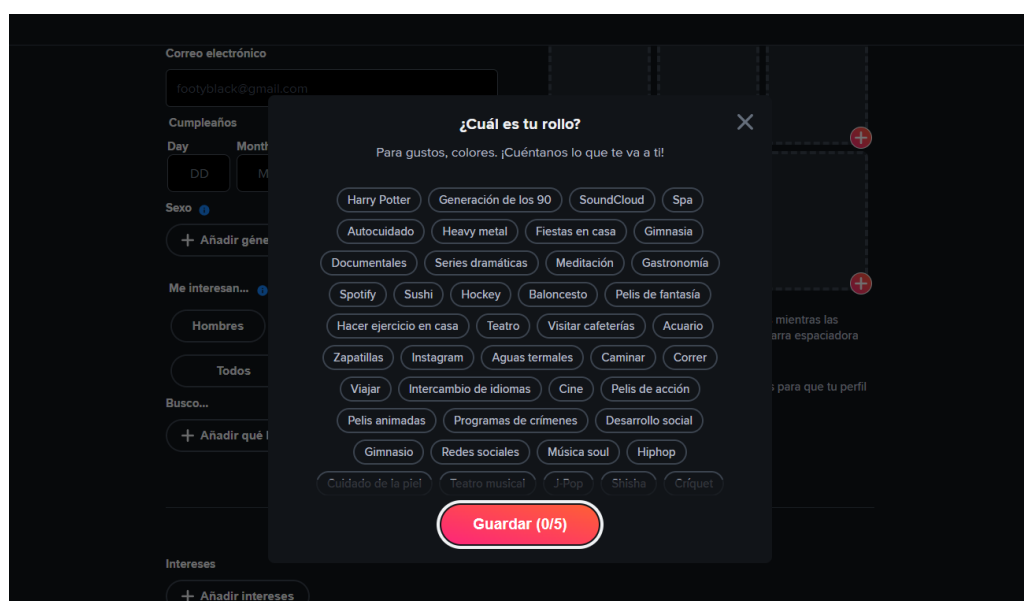


Figura 2.5: Selección opcional de intereses en Tinder durante la creación del perfil.

2.1.2.2. Spotify

Spotify [5] es una plataforma de música en streaming que destaca por su sistema de recomendaciones personalizadas. A partir del historial de escucha, listas guardadas o canciones favoritas, sugiere contenido ajustado a los gustos del usuario.

En la Figura 2.6 se muestran ejemplos de *Mixes diarios*, listas generadas automáticamente en función de preferencias individuales. De forma similar, la Figura 2.7 muestra una selección de vídeos musicales relevantes para el perfil del usuario.

Aunque Spotify no empareja personas, su modelo sirve de referencia para esta plataforma, especialmente en dos aspectos: mostrar resultados filtrados y relevantes en lugar de listados genéricos, y calcular una **afinidad cuantificada** entre usuarios y contenido, concepto que se traslada aquí al emparejamiento entre profesionales y proyectos.

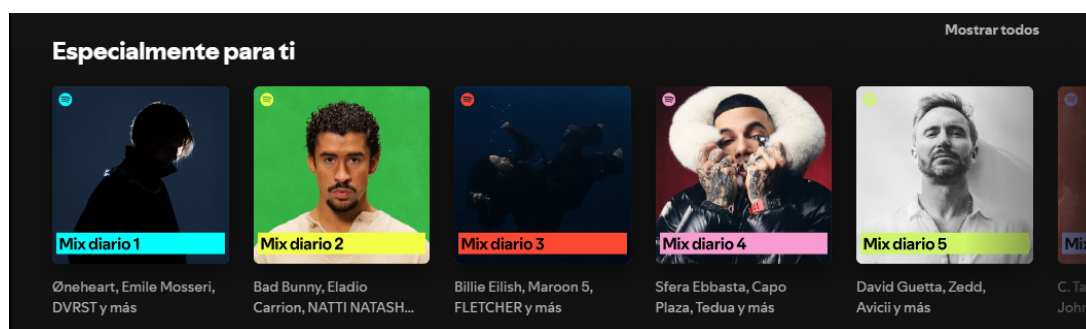


Figura 2.6: Recomendaciones musicales personalizadas mediante mezclas diarias en Spotify.

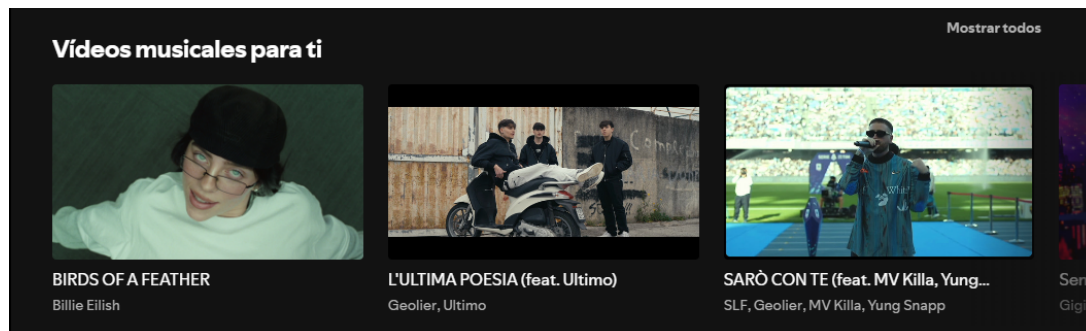


Figura 2.7: Selección de vídeos musicales adaptados a los gustos del usuario.

2.1.2.3. Shapr

Shapr [6] es una aplicación de networking profesional que emplea un sistema de emparejamiento entre usuarios a partir de información como el área profesional, ubicación, experiencia, año académico e intereses generales. A diario sugiere un número limitado de perfiles o empleos potencialmente afines, ofreciendo una experiencia sencilla y enfocada en generar conexiones útiles.

Aunque está centrada en el ámbito profesional, Shapr no funciona como un portal de empleo tradicional, sino que utiliza una lógica de emparejamiento basada en sugerencias automatizadas y validación mutua, similar a lo que ocurre en aplicaciones como Tinder.

Por este motivo, se ha incluido en este bloque, centrado en sistemas de coincidencia, y no en el anterior, dedicado a plataformas orientadas a la gestión del talento.

En la Figura 2.8 se muestra su sección de ofertas laborales, donde se presentan prácticas o puestos adaptados al perfil del usuario. Este filtrado no requiere una búsqueda activa ni ajustes manuales por parte del usuario, como sucede en plataformas como LinkedIn. En su lugar, el sistema interpreta automáticamente los datos proporcionados, como el grado, la ubicación o el año académico. Por ejemplo, si el usuario indica que está cursando el último año de carrera, es habitual que se prioricen ofertas de prácticas o contratos de formación.

A diferencia de las plataformas anteriores, Shapr sí está enfocada al entorno profesional, lo que la convierte en una referencia especialmente relevante para este trabajo. La plataforma propuesta recoge esa misma idea de sencillez en la presentación, pero la aplica al sistema ya comentado, basado en competencias y coincidencias automáticas, acompañado de un indicador de afinidad.

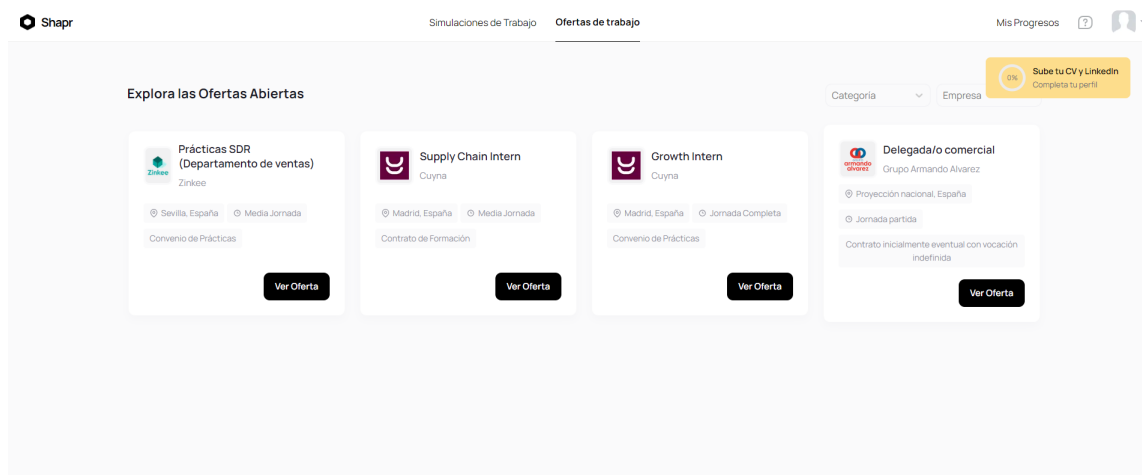


Figura 2.8: Listado de ofertas laborales en Shapr ajustadas al perfil del usuario.

2.1.2.4. Conclusiones

Las aplicaciones analizadas en este bloque comparten la idea de emparejamiento por coincidencia, aunque con enfoques distintos. **Tinder** introduce la lógica del *interés mutuo* a partir de sugerencias generales, mientras que **Spotify** aporta el concepto de **recomendación personalizada** y afinidad basada en comportamientos previos.

Shapr, a diferencia de las anteriores, está orientada al ámbito profesional y muestra ofertas adaptadas a la información del usuario, aunque sin usar competencias ni mostrar un porcentaje de coincidencia. Su modelo sirve de inspiración por su **presentación clara y filtrado automatizado**.

Estas soluciones aportan **ideas clave**, pero ninguna cubre el emparejamiento profesional desde una perspectiva basada en competencias y afinidad técnica. Por ello, se plantea la creación de una **plataforma específica** como la propuesta en este [TFG](#).

2.2. Evaluación de tecnologías

Capítulo 3

Requisitos, especificaciones, planificación, coste, riesgos y viabilidad

3.1. Requisitos

3.1.1. Requisitos funcionales

3.1.2. Requisitos no funcionales

Se debe optar por formular los requisitos de forma que se pueda conocer si se han alcanzado o no a la finalización del proyecto. Por ejemplo, es difícil valorar si el siguiente requisito funcional se alcanza o no: *El sistema debe retornar una respuesta en un tiempo razonable cuando tenga muchos usuarios concurrentes*. ¿Cuánto es un tiempo razonable?, ¿cuántos son muchos usuarios?. Sin embargo, si se formula de este otro modo: *El sistema debe retornar una respuesta en menos de un segundo cuando tenga 200 usuarios concurrentes*, es fácil comprobar si se ha alcanzado ejecutando un plan de pruebas, por ejemplo con JMeter.

3.2. Especificaciones

Especificación del proyecto a partir de los requisitos. Una vez se han definido los requisitos funcionales y no funcionales del sistema, se procede a especificar el sistema a partir de ellos. En este apartado se muestran características del sistema, sin entrar en detalles de diseño e implementación, que permiten entender el problema al que nos enfrentamos pero no como lo vamos a afrontar (etapa de diseño). Se recomienda confeccionar una arquitectura de referencia, o imagen no técnica (más bien comercial) que ilustre los actores y aplicaciones principales que conforman el sistema.

3.3. Planificación y estimación de costes

Describir el tipo de metodología de desarrollo que se va a utilizar (cascada, ágil, etc). Tareas a realizar, estimación de la duración de las tareas, y distribución temporal (por ejemplo con un diagrama de Gantt).

Enumerar las tareas a realizar, estimación de la duración de las tareas, y distribución temporal (por ejemplo con un diagrama de Gantt).

Costes de personal (teniendo en cuenta los costes de seguridad social), de hardware (imputando solo la duración del proyecto y teniendo en cuenta que los equipos se amortizan en 3 o 4 años) y/o de software. Además, hay que añadir costes indirectos.

3.4. Riesgos

Identificación de los riesgos que pueden aparecer durante el desarrollo del proyecto, su probabilidad de ocurrencia, su impacto en el proyecto y las medidas que se podrían adoptar para mitigarlos.

3.5. Viabilidad

En este apartado, dependiendo de la naturaleza del proyecto, se debería analizar la viabilidad técnica y la viabilidad económica. Para la viabilidad técnica hay que analizar si los recursos necesarios (herramientas, conocimientos, experiencia, etc) para llevar a cabo el proyecto permiten realizarlo en el tiempo previsto. En cuanto a la viabilidad económica hay que evaluar si el proyecto será rentable cuando esté operativo.

Capítulo 4

Análisis

4.1. Diagrama de casos de uso

Identificación de actores y diagramas de casos de uso asociados a los requisitos funcionales.

4.2. Diagrama de secuencia

Para los casos de uso más importantes y complejos se deben especificar los diagramas de secuencia que describan el comportamiento de los objetos que intervienen en el caso de uso.

4.3. Diagrama de clases de primer nivel

Al final del análisis se debe presentar un diagrama de clases de primer nivel que muestre las clases principales del sistema y sus relaciones. Para cada una de las clases que tengan un estado complejo o relevante, se debe especificar el diagrama de estados asociado o el diagrama de actividades en el que se involucra la clase.

Capítulo 5

Diseño

5.1. Arquitectura de componentes

A partir de la especificación realizada en la sección 3.2 del Capítulo 3 y la tecnología seleccionada en la sección 2.2 del Capítulo 2, se debe construir el diagrama de componentes UML que represente la arquitectura de referencia del sistema. En este diagrama se deben identificar los componentes principales del sistema y las relaciones entre ellos, pasando a detallar los subcomponentes de cada uno de ellos en secciones posteriores.

5.2. Despliegue del sistema

En esta sección se debe presentar el diagrama de despliegue del sistema, que muestre la arquitectura física del sistema y cómo se distribuyen los componentes en los nodos físicos o virtuales que lo componen. En otras palabras, para cada componente principal del sistema se identifica en que servidor físico o virtual se va a ejecutar.

Se puede definir un despliegue de desarrollo (el despliegue que se tiene en el portátil mientras se trabaja), uno de producción (despliegue final o real) y uno de pruebas (lo más parecido posible al real).

5.3. Modelo de datos

En este caso se puede especificar el diagrama de clases final del diseño atendiendo al lenguaje de programación y a la tecnología seleccionada. Además será necesario, como mínimo, especificar el modelo físico de datos, en el lenguaje SQL propio de la base de datos relacional seleccionada, o el modelo de documentos, si se utiliza una base de datos NoSQL.

Capítulo 6

Implementación y pruebas

Las secciones presentadas son orientativas y no representan necesariamente la organización que debe tener este capítulo.

6.1. Implementación

Presentar cómo se ha organizado el desarrollo de los proyectos (capturas del IDE), trozos de código relevantes, cómo han quedado implementadas las interfaces gráficas de usuario, etc.

6.2. Pruebas unitarias

Descripción de las pruebas que se han llevado a cabo para comprobar que el código desarrollado es correcto (JUnit, etc).

6.3. Pruebas funcionales

Descripción de las pruebas que se han llevado a cabo para comprobar que los casos de uso identificados funcionan correctamente.

6.4. Pruebas de rendimiento

Descripción de las pruebas de estrés realizadas para comprobar los tiempos de respuesta de la aplicación (según figuren en los requisitos).

6.5. Pruebas de usabilidad

Descripción de las pruebas que se han llevado a cabo con usuarios para determinar el nivel de usabilidad de la aplicación (que se hayan recogido en los requisitos).

6.6. Pruebas de seguridad

Descripción de las pruebas realizadas para comprobar que se cumplen las restricciones de autenticación y de autorización que se han descrito en los requisitos.

Capítulo 7

Conclusiones

7.1. Revisión de costes

Al finalizar el proyecto hay que ver en qué fases de la ejecución nos hemos desviado, explicar los motivos y calcular el coste real con estos ajustes.

7.2. Conclusiones

7.3. Trabajo futuro

Capítulo 8

Bibliografía

- [1] LinkedIn Corporation, *LinkedIn*, Sitio web oficial de LinkedIn, red profesional para la búsqueda de empleo y gestión de talento, 2025. visitado 18 de abr. de 2025. dirección: <https://www.linkedin.com> (citado en páginas 25, 26).
- [2] Tinder Inc., *Tinder*, Sitio web oficial de Tinder, aplicación de emparejamiento basada en afinidad, 2025. visitado 18 de abr. de 2025. dirección: <https://www.tinder.com> (citado en páginas 25, 29, 30).
- [3] InfoJobs S.L., *InfoJobs*, Sitio web oficial de InfoJobs, plataforma de búsqueda de empleo, 2025. visitado 18 de abr. de 2025. dirección: <https://www.infojobs.net> (citado en páginas 26, 27).
- [4] Yobalia S.L., *Yobalia*, Sitio web oficial de Yobalia, plataforma especializada en ofertas de empleo para promociones, eventos y azafatas, 2025. visitado 18 de abr. de 2025. dirección: <https://www.yobalia.com> (citado en páginas 26, 28).
- [5] Spotify Technology S.A., *Spotify*, Sitio web oficial de Spotify, plataforma de streaming de música, 2025. visitado 18 de abr. de 2025. dirección: <https://www.spotify.com> (citado en páginas 29, 31).
- [6] Shapr Inc., *Shapr*, Sitio web oficial de Shapr, aplicación de networking profesional, 2025. visitado 18 de abr. de 2025. dirección: <https://www.shapr.co> (citado en páginas 30, 31).
- [7] Raúl Peña Ortiz, “Accurate workload design for web performance evaluation,” Tesis doctoral, Universitat Politècnica de València, ene. de 2013, ISBN: 9788490480250. DOI: [10.4995/thesis/10251/21054](https://doi.org/10.4995/thesis/10251/21054) (citado en página 48).
- [8] Frank Mittelbach y Ulrike Fischer, *The L^AT_EX Companion*, 3.^a edición. Addison-Wesley, 2023, ISBN: 978-0138166489 (citado en página 55).
- [9] H. Frank Cervone, “Getting Started with Cloud Computing: A LITA Guide,” en Edward M. Corrado y Heather Lea Moulaison, edición. Neal-Schuman Publishers, Inc, 2011, capítulo Cloud Computing: Pros and Cons, páginas 30-31, ISBN: 978-1-55570-749-1 (citado en página 55).
- [10] Roy Thomas Fielding y Richard N. Taylor, “Principled Design of the Modern Web Architecture,” *ACM Transactions on Internet Technology*, volumen 2, número 2, páginas 115-150, mayo de 2002, ISSN: 1533-5399. DOI: [10.1145/514183.514185](https://doi.org/10.1145/514183.514185) (citado en página 55).

- [11] Roy Thomas Fielding y Richard N. Taylor, “Principled Design of the Modern Web Architecture,” en *Proceedings of the 22nd International Conference on Software Engineering*, época ICSE '00, Limerick, Ireland: Association for Computing Machinery, jun. de 2000, páginas 407-416, ISBN: 1581132069. DOI: [10.1145/337180.337228](https://doi.org/10.1145/337180.337228) (citado en página 55).
- [12] Andoni Salcedo-Navarro, Raúl Peña-Ortiz, José M. Claver, Miguel Garcia-Pineda y Juan Gutiérrez-Aguado, “Cloud-Native GPU-Enabled Architecture for Parallel Video Encoding,” en *Euro-Par 2024: Parallel Processing*. Springer Nature Switzerland, ago. de 2024, páginas 327-341, ISBN: 9783031695834. DOI: [10.1007/978-3-031-69583-4_23](https://doi.org/10.1007/978-3-031-69583-4_23) (citado en página 55).
- [13] Roy Thomas Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” Tesis doctoral, University of California, Irvine, 2000. dirección: https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf (citado en página 55).
- [14] Peter Mell y Tim Grance, “The NIST Definition of Cloud Computing,” NIST Special Publication 800-145, 2011. DOI: [10.6028/nist.sp.800-145](https://doi.org/10.6028/nist.sp.800-145) (citado en página 55).
- [15] Alexey Melnikov, “The WebSocket Protocol,” Internet Engineering Task Force (IETF), Standard, dic. de 2011. dirección: <https://tools.ietf.org/html/rfc6455> (citado en página 55).
- [16] Luis Thuillier Larena, “Sistema de control de aforos dinámico mediante dispositivos IoT,” Trabajo Fin de Grado, Escola Tècnica Superior d’Enginyeria - Universitat de València, 2021 (citado en página 55).
- [17] MA Awad, “A comparison between agile and traditional software development methodologies,” Trabajo Fin de Máster, University of Western Australia, 2005. dirección: <https://www.academia.edu/download/58993716/10.1.1.464.609020190422-13963-j0ju8a.pdf> (citado en página 55).
- [18] Django Software Foundation, “Django Documentation,” Manual Release 4.1.4, jul. de 2023. dirección: <https://buildmedia.readthedocs.org/media/pdf/django/4.1.x/django.pdf> (citado en página 55).
- [19] Microsoft, *Visual Studio Code*, 2023. visitado nov. de 2023. dirección: <https://code.visualstudio.com> (citado en página 55).

Apéndice A

Glosarios

A.1. Acrónimos y abreviaciones

SGBD

Sistema de Gestión de Base de Datos. [52](#)

TFG

Trabajo Fin de Grado. [5](#), [7](#), [9](#), [11](#), [25](#), [28](#), [29](#), [32](#), [55](#)

TFM

Trabajo Fin de Máster. [52](#), [55](#)

A.2. Términos

Backend

Parte de una aplicación web que se encarga de la lógica del servidor y la gestión de datos. [26](#)

Frontend

Parte de una aplicación web que interactúa directamente con el usuario. [26](#)

Full Stack

Perfil de desarrollador que trabaja tanto en el desarrollo del lado cliente (frontend) como en el servidor (backend) de una aplicación web. [26](#)

Match

Coincidencia entre dos perfiles en un sistema de emparejamiento digital, determinada por afinidad entre características, intereses o competencias. Puede generarse de forma automática o mediante la validación mutua de las partes implicadas, y permite habilitar la interacción entre ellas. [30](#)

Middleware

Software que se encuentra entre el sistema operativo y las aplicaciones que funcionan sobre él. [52](#)

Sistema de recomendaciones

Mecanismo que sugiere elementos (proyectos o usuarios) en función de similitudes, coincidencias o preferencias. [26](#)

Apéndice B

Ejemplos del lenguaje de marcado L^AT_EX

B.1. Algunos ejemplos básicos de uso de L^AT_EX

Texto en el párrafo 1.

Texto en el párrafo 2.

Texto en el párrafo 3.

- Consideración 1
- Consideración 2

1. Punto 1

2. Punto 2

A continuación se muestra una ecuación:

$$\int_0^1 \frac{1}{x^2 + 1} dx$$

B.2. Figuras y subfiguras

Podemos incluir imágenes en formato: png, pdf o jpg.

En la Figura B.1 se muestra un diagrama¹, mientras que la Figura B.2 muestra varios diagramas en mosaico, de manera que puedo referirme al segundo diagrama como Figura B.2b.

¹Realizado con <https://www.yworks.com/products/yed>

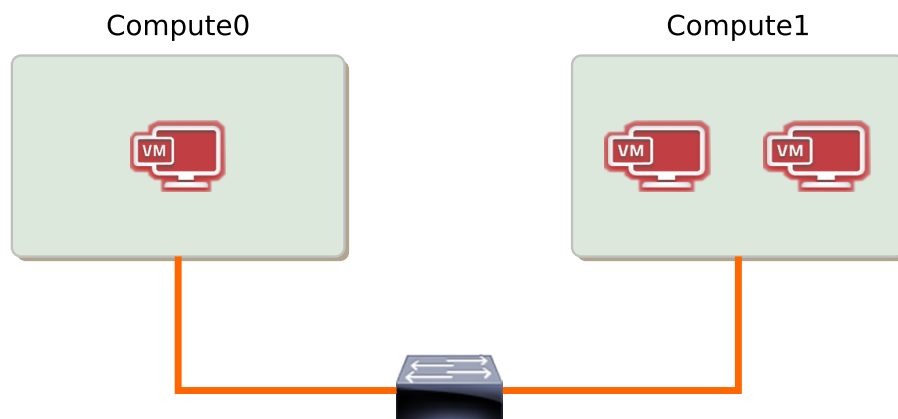


Figura B.1: Esta es una figura que latex decide donde colocar (floating) en el documento.

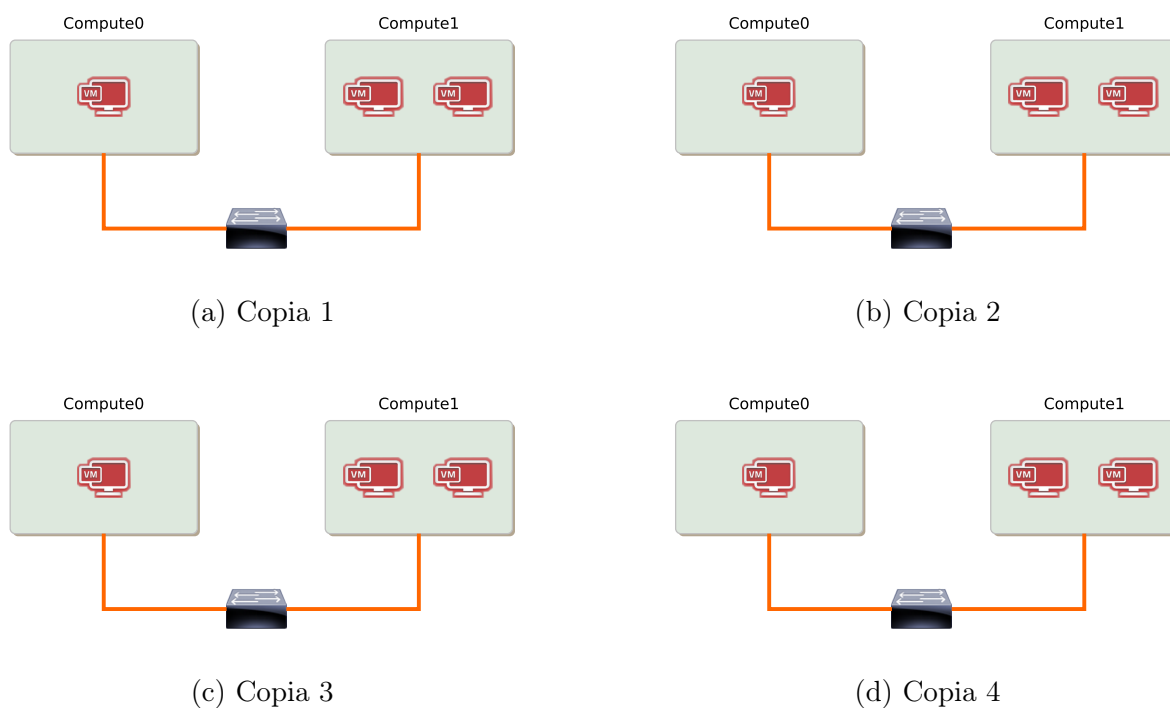


Figura B.2: Los cuatro diagramas en mosaico

B.3. Tablas

La Tabla B.1 es un ejemplo de una tabla.

Columna 1	Columna 2
1	2

Tabla B.1: Ejemplo de tabla 1

Existen también tablas más complejas que requieren un mayor control de la posición de los elementos, como la Tabla B.2 que ha sido extraída de [7], o de varias páginas, como la Tabla B.3.

GENERATOR FEATU./CAPAB.	GROUP I					GROUP II			GROUP III								
	WebStone	SPECweb	SURGE	Web Polygraph	TPC-W	LoadRunner	WebLOAD	JMeter	S-Clients	WebJamma	Deluge	HAMMERHEAD 2	PTester	Siege	HTTPERF	Autobench	
	★	★	★	★	★	★	★	★									
	★	★	★	★		★	★										
	★	☆		☆	★	★	★	★		★		☆			☆		
	★	★			★	★	★	★									
		★			★	★	★	★									
						★	★	★						★		★	
									★				☆				
		★		★	★	★	★	★	★	★	★	★	★	★	★	★	★
						★	★	★									
		★		★	★	★	★	★				★	★	★	★	★	★
	User's Dynamism					★	★	★	★	★	★	★	★	★	★	★	★

★ Full support ☆ Partial support

Tabla B.2: Web workload generators and grade in which main features are fulfilled

Tabla B.3: Ejemplo de tabla muy larga para una página

[illegible]

Tabla B.3 – continued from previous page

[illegible]

B.4. Listados

Para generar el fichero PDF, podemos usar los comandos del Listado B.1.



```
1 pdflatex main.tex
2 bibtex main
3 pdflatex main.tex
```

Listado B.1: Comandos bash para generar el fichero PDF

También se puede usar `arara` que automáticamente regenera la bibliografía, ver los comandos del Listado B.2.

```
1 | arara main.tex
```

Listado B.2: Comandos bash para generar el fichero PDF con arara

También se pueden incluir listados de otros lenguajes como Python, ver Listado B.3, Java, ver Listado B.4, HTML , ver Listado B.5 o CSS , ver Listado B.6.

```
1 | def hello():
2 |     print("Hello World!")
```

Listado B.3: Ejemplo de listado en Python

```
1 | public class HelloWorld {
2 |     public static void main(String[] args) {
3 |         System.out.println("Hello, World");
4 |     }
5 | }
```

Listado B.4: Ejemplo de listado en Java

```
1 | <!DOCTYPE html>
2 | <html>
3 | <head>
4 | <title>Page Title</title>
5 | </head>
6 | <body>
7 |
8 | <h1>This is a Heading</h1>
9 | <p>This is a paragraph.</p>
10 |
11 | </body>
12 | </html>
```

Listado B.5: Ejemplo de listado en HTML 

```
1 | body {
2 |     background-color: lightblue;
3 | }
4 |
5 | h1 {
6 |     color: white;
7 |     text-align: center;
8 | }
9 |
10 | p {
11 |     font-family: verdana;
12 |     font-size: 20px;
13 | }
```

Listado B.6: Ejemplo de listado en CSS 

B.5. Acrónimos y términos

L^AT_EX tiene un paquete de glosario de términos y acrónimos que permite definir términos y acrónimos y usarlos en el texto. Lo que al principio puede parecer engorroso, una vez se usa, es muy útil.

Usamos por primera vez el acrónimo Trabajo Fin de Máster (TFM) y luego volvemos a usar el acrónimo TFM, que ahora aparece abreviado, y por primera vez el término *Middleware*.

Tenemos una opción especial para la forma plural de los acrónimos, como Sistema de Gestión de Base de Datos (SSGGBDD), aunque luego se usen en singular, SGBD.

Podemos forzar que aparezca el término completo, Trabajo Fin de Grado, abreviado, TFG, o ambos, Trabajo Fin de Grado (TFG). Aunque en estos casos no nos lleva a la tabla de acrónimos y términos.

El fichero `acronimos-terminos.tex` contiene ejemplos de definición de acrónimos y términos.

Apéndice C

Ejemplos de uso de $\text{BIB}\text{T}_{\text{E}}\text{X}$ como gestor de bibliografía

Este apéndice muestra ejemplos de uso de $\text{BIB}\text{T}_{\text{E}}\text{X}$ como gestor de bibliografía.

C.1. Ejemplos de referencias formales bien valoradas

- Un libro [8] o un capítulo de libro [9].
- Un artículo de revista [10].
- Una publicación en una conferencia [11] o una publicación de conferencia que forma parte de una colección [12].
- Una tesis doctoral [13].
- Un informe técnico específico del NIST [14].

C.2. Ejemplos de referencias con valoración intermedia

- Un informe técnico genérico [15].
- Un TFG [16] o un TFM [17].

C.3. Ejemplos de referencias con valoración baja a evitar en la medida de lo posible

- Un manual online [18].
- Una página web [19].