Documentación del proyecto
SkillWatch

1.- Introducción

En este proyecto abordaremos una aplicación web para llevar una gestión del alumnado de prácticas en centros de trabajo, en el cual haremos uso de las distintas tecnologías impartidas a lo largo del curso (frameworks como Laravel y Angular parra el desarrollo del back y front, HTML5, CSS, bootstrap para darle la estructura y estilo a la página web, Xampp para el servidor local, etc).

2.- Estudio de viabilidad

En este apartado evaluaremos la viabilidad del proyecto, abordando distintos puntos que detallaremos a continuación:

- **Análisis de mercado**: debido a la gran capacidad informática de la actualidad, el uso de apps o webs que faciliten el trabajo en cierta medida está muy demandado, para ser así más productivo. Por ello, es una gran oportunidad para implementar una web que facilite la gestión y seguimiento del alumnado en centros de trabajo.
- **Análisis financiero**: el centro donde se implantará esta solución dispone de conexión a internet y de equipos informáticos que soportan a la perfección los requisitos mínimos, por lo que no sería necesario realizar una inversión inicial.
- **Análisis técnico**: al disponer de los requisitos mínimos para la implantación del proyecto (equipos informáticos y conexión a internet) solo sería necesario configurar los equipos que vayan a hacer uso de esta solución, y dar la documentación necesaria al usuario para informarle de las funcionalidades disponibles.
- **Análisis de competencia**: aunque hoy día se están implantando muchas soluciones web a casos como este, nuestra solución web ofrece un estilo minimalista, simple, funcional y con muchas posibles configuraciones para mejorar la experiencia del usuario, por lo que hace que se diferencie del resto.

2.1.- Descripción del Sistema Actual

Actualmente, se usa un sistema muy rudimentario y manual para llevar la gestión y seguimiento del alumnado, como el uso de (excel, word, etc).

Por esto es necesario actualizar el método con el que se lleva a cabo la gestión y seguimiento del alumnado.

2.2.- Descripción del Sistema Nuevo

Debido al sistema actual tan rudimentario surge este proyecto donde se automatizan procesos y se agilizarán acciones para así facilitar esta tarea.

Se tratará de sustituir el uso de excel, word y demás sistemas de gestión de información por un solo sistema unificado: Skillwatch. Es una herramienta diseñada para que no sea necesario un gran conocimiento en informática, muy intuitiva. Facilitará a los profesores la gestión de alumnos en prácticas, las distintas empresas y sedes donde se encuentran, así como la información personal de éstos.

2.3.- Identificación de Requisitos del Sistema

2.3.1.- Requisitos de la información

Será necesario almacenar datos de:

- **El alumno en prácticas** : será necesario distinguir si el alumno se encuentra en el ciclo de DAW (Desarrollo de Aplicaciones Web) o DAM (Desarrollo de Aplicaciones Multiplataforma). Además de esto, será necesario almacenar su nombre, apellido, su correo electrónico y su contraseña para ingresar a la plataforma.

  Se consideran todos los campos obligatorios para poder llevar un correcto seguimiento del alumnado.

- **Las ofertas** : enfocada a las empresas para que transmitan su necesidad de cubrir vacantes al centro. Será necesario distinguir si el alumno requerido se encuentra en el ciclo de DAW o DAM. Además de esto, será necesario almacenar el estado de la oferta (aceptada, denegada o en curso) y el lugar donde tendrá lugar el desarrollo de las prácticas.

Se consideran todos los campos obligatorios para poder llevar un correcto seguimiento de las ofertas existentes con el fin de adecuarse lo mejor posible a las necesidades de la empresa y del alumnado.

- **Las empresas** : enfocada a llevar el control de las empresas exitentes. Será necesario almacenar el nombre de la empresa, el número de teléfono y el email de la compañía.

  Se consideran todos los campos obligatorios para poder llevar un correcto seguimiento de las empresas existentes para facilitar el contacto con estas en caso de algún problema con respecto al alumnado.

- **Sedes de las empresas** : enfocada a determinar las distintas sedes de las empresas. Será necesario almacenar el nombre de la sede, enlazarlo con la empresa a la que pertenece dicha sede y los alumnos que realizarán sus prácticas en esa sede.

  Se consideran todos los campos obligatorios para poder llevar un correcto seguimiento de las sedes existentes para tener clara la distribución del alumnado.

El volumen de información estimado, teniendo en cuenta todo lo anterior, será de aproximadamente 9600 bytes teniendo en cuenta una media de 30 alumnos, 5 profesores, 25 empresas y 50 sedes.

2.3.2.- Requisitos de funcionales:

Método	Ruta	Descripción
POST	http://127.0.0.1:8000/api/register	Para registrar un usuario
POST	http://127.0.0.1:8000/api/login	Para iniciar sesión
POST	http://127.0.0.1:8000/api/get-user	Devuelve los datos del usuario
POST	http://127.0.0.1:8000/api/logout	Para desloguear al usuario
POST	http://127.0.0.1:8000/api/companies	Para crear una empresa
GET	http://127.0.0.1:8000/api/companies/{company_id}	Muestra los datos de una empresa en específico
GET	http://127.0.0.1:8000/api/companies	Muestra un listado con todas las empresas
PUT	http://127.0.0.1:8000/api/companies/{company_id}	Actualiza una empresa
GET	http://127.0.0.1:8000/api/company_headquarters/{company_id}	Muestra las sedes de una empresa
POST	http://127.0.0.1:8000/api/headquarter	Crea una sede
PUT	http://127.0.0.1:8000/api/headquarter	Actualiza una sede
DELETE	http://127.0.0.1:8000/api/headquarter/2	Elimina una sede
GET	http://127.0.0.1:8000/api/offers	Devuelve un listado de todas las ofertas
POST	http://127.0.0.1:8000/api/offers	Para crear una oferta
PUT	http://127.0.0.1:8000/api/offers	Para actualizar una oferta
DELETE	http://127.0.0.1:8000/api/offers/{offer_id}	Para eliminar una oferta

2.4.- Descripción de la solución

La solución web ofrece:

- Un diseño minimalista diseñado con HTML5, bootstrap y CSS.
- Un back-end desarrollado con Laravel
- Un front-end desarrollado con Angular
- Git hub como controlador de versiones.
- MySQL para la gestión y creación de la base de datos.
- Xampp para alojar la web en un servidor local (de esta manera no se necesita de conexión a internet para acceder a la web).

Para elegir estas tecnologías se ha tenido en cuenta su eficacia y su mantenimiento el tiempo.

2.5.- Planificación del proyecto

2.5.1.- Equipo de trabajo

El desarrollo del proyecto se llevará a cabo con el siguiente equipo de trabajo:

- **Desarrollador/es front-end:**
    - Álvaro Vega Peso
    - Enrique Sánchez Gómez
- **Desarrollador/es back-end:**
    - Álvaro Vega Peso
    - Jesús Domínguez Domínguez
    - Belén Márquez Bonilla
- **Diseñador/es:**
    - Mario Vega Peso
- **Testers:**
    - Claudia Sánchez Rueda

    - Gema Peso Laguna
- **Manager del proyecto:**
    - Álvaro Vega Peso

2.5.2.- Planificación temporal

El proyecto cuenta con la planificación que detallaremos a continuación:

- Actividades: el proyecto permite realizar las actividades expuestas en el enunciado del proyecto integrados
- Timpo estimado:
    - Planificación de los requisitos (usuarios necesarios, requisitos técnicos, planificación de la base de datos, diseño, etc) 2 semanas
    - Desarrollo de la base de datos 1 semana
    - Desarrollo del back-end 2 semanas
    - Desarrollo del diseño 2 semanas
    - Desarrollo del front-end 2 semanas
    - Test sobre el sistema 1 semana

2.6.- Estudio del coste del proyecto

El coste del proyecto es mínimo ya que no es necesario un servidor web, la web se aloja en un servidor local y se disponen de equipos que soportan los requisitos mínimos.

Lo único necesario sería realizar una guía de usuario donde se especifiquen las distintas funcionalidades del sistema y el funcionamiento de los flujos.

También sería necesario un grupo de técnicos que den soporte a las diferentes incidencias que se puedan ocasionar y den un mantenimiento a la web.

3.- Análisis del Sistema de Información

3.1.- Identificación del entorno tecnológico

Para el desarrollo de la aplicación web hemos utilizado las siguientes tecnologías:

- Para la base de datos hemos usado MySQL
- Hemos usado Xampp para desplegar la web en un servidor local
- Hemos usado el framework Laravel en su versión 8 para el desarrollo del back-end ya que permite trabajar de manera ordenada y facilita el mantenimiento del back-end
    - Para que funcione adecuadamente Laravel hemos tenido que instalar composer y git (para llevar un control del proyecto de las versiones del mismo)
- Hemos usado el framework Angular para el desarrollo del frony-end, este framework facilita en gran medida el desarrollo de la web al igual que su mantenimiento ya que funciona por componentes.
- Para el diseño de la web hemos usado HTML5 para la estructura y bootstrap junto con CSS para darle estilo.

3.2.- Modelado de datos

3.2.1.- Esquema de la base de datos

Almacenaremos la información en las siguientes tablas, que contienen los datos que detallaremos a continuación.

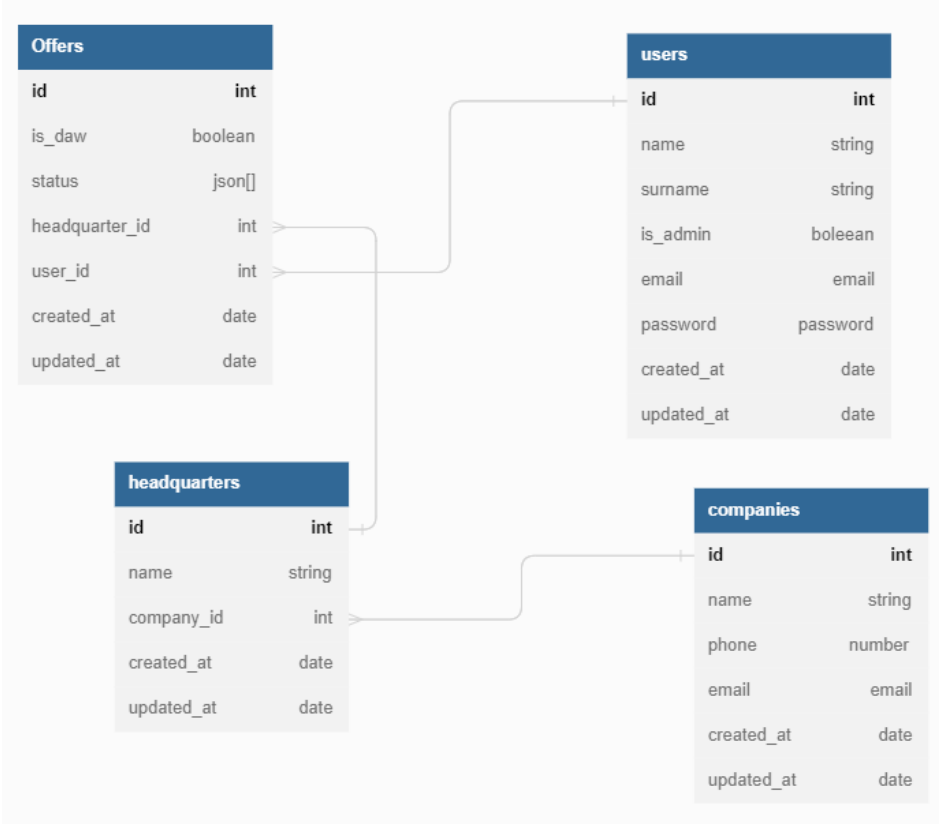

Figura 1. Relaciones entre tablas.

- **Offers**: esta tabla está destinada a almacenar la información de las ofertas que lanzan las empresas, la cual cuenta con los siguientes datos:
    - Id: número de identificación único para cada oferta (tipo number).
    - is_daw: campo para llevar un control de cuantos alumnos de DAW solicita cada empresa (tipo number).
    - status: campo en el que se almacena la información del estado de la oferta: en curso (tipo Json) y puede adquirir los siguientes valores: en curso , aceptado y rechazado.

- headquarter_id: este campo establece una distinción entre las sedes de las empresas. Enlaza la tabla offers con la tabla headquarters (tipo int, una sede puede tener varias ofertas)
- user_id: campo que hace referencia al id del usuario para enlazar la tabla offers con la tabla user_id (tipo int, un usuario puede tener varias ofertas).
- created_at: campo donde almacenaremos la fecha de creación de las ofertas (tipo date).
- updated_at: campo donde almacenaremos la fecha de actualización de las ofertas (tipo date).

- **Users**: esta tabla está destinada a almacenar la información de los usuarios, la cual cuenta con los siguientes datos:
  - id: número de identificación único para cada usuario (tipo number).
  - name: donde almacenaremos el nombre del alumno (tipo string).
  - surname: donde almacenaremos el apellido del alumno (tipo string).
  - is_admin: campo que utilizamos para determinar si un usuario actua o no como administrador (tipo boolean, los alumno al no actuar como admin tienen este campo a false por defecto).
  - email: emai de contacto del alumno también lo usaremos para hacer login (tipo string).
  - password: en este campo almacenaremos la contraseña del usuario, la cual encriptaremos por motivos de seguridad (tipo string)
  - created_at: campo donde almacenaremos la fecha de creación del usuario (tipo date).
  - updated_at: campo donde almacenaremos la fecha de actualización del usuario (tipo date)

- **Headquarters**: esta tabla está destinada a almacenar la información de las sedes de las que dispone cada empresa, la cual cuenta con los siguientes datos:

- id: número de identificación único para cada sede (tipo number).
- name: donde almacenaremos el nombre de la sede (tipo string).
- company_id: campo que hace referencia al id de la empresa para enlazar la tabla companys con la tabla headquarters (tipo number, una empresa puede tener varias sedes).
- created_at: campo donde almacenaremos la fecha de creación de las sedes (tipo date).
- updated_at: campo donde almacenaremos la fecha de actualización de las sedes (tipo date).

- **Companies**: esta tabla está destinada a almacenar la información de las empresas, la cual cuenta con los siguientes datos:
  - id: número de identificación único para cada empresa (tipo int).
  - name: donde almacenaremos el nombre de la empresa (tipo string).
  - phone: campo donde almacenaremos el número de contacto de la empresa (tipo number).
  - email: almacena el emai de contacto de la empresa (tipo email).
  - created_at: campo donde almacenaremos la fecha de creación de la empresa (tipo date).
  - updated_at: campo donde almacenaremos la fecha de actualización de la empresa (tipo date).

3.2.3.- Datos de prueba

(archivo adjunto)

3.3.- Identificación de los usuarios participantes y finales

El planteamiento de la web ofrece dos perfiles:

- Profesor: actúa como administrador teniendo permisos para crear/actualizar/eliminar datos de cualquiera de las tablas planteadas.
- Alumno: perfil enfocado en visualizar datos, sus candidaturas y el estado de las mismas, además de subir el CV en formato PDF.

3.4.- Identificación de subsistemas de análisis

A continuación detallaremos los diferentes subsistemas que componen nuestra aplicación web:

- **Gestión de empresas (companies):** la realiza el perfil de profesor como administrador, esta interactúa con las ofertas y a su vez con las sedes (headquarters) mediante una relación 1-N (una empresa puede tener varias sedes).
- **Gestión de usuarios (users):** la realiza el perfil de professor (profesor) como administrador del sistema, esta interactúa con las ofertas (offers) mediante una relación 1-N (un usuario puede tener varias ofertas).
- **Gestión de sedes (headquarters):** la realiza el perfil de profesor, las sedes se relacionan con las las empresas (companies) y con ofertas (offers), ambas con mediante una relación 1-N (una sede puede tener varias ofertas y una empresa puede tener varias sedes).
- **Gestión de ofertas (offers):** la realiza el perfil de profesor, se enlaza con las sedes (headquarters) mediante una relación 1-N (una sede puede tener varias ofertas).

3.5.- Establecimiento de requisitos

A continuación describiremos las diferentes funcionalidades que ofrecen los diferentes perfiles disponibles en nuestra solución web:

Los profesores presentan las siguientes funcionalidades:

- Se inicia con un register que solicita los siguientes campos:
    - Nombre
    - Apellidos
    - Email
    - Contraseña
- Login con los siguientes campos:
    - Email
    - Contraseña
- Este perfil actua como administrador por lo que tiene permisos para crear/modificar/eliminar cualquier dato del sistema

Detallaremos el funcionamiento de los flujos del sistema correspondiente a este perfil:

- Una empresa puede ofertar varios puestos de prácticas
- Una oferta se corresponde con una sede
- Una empresa puede tener varias sedes
- Un usuario alumno puede tener varias ofertas

El perfil de usuario ofrece las siguientes funcionalidades:

- Register en el cual se solicita la siguiente información:
    - Nombre
    - Apellidos
    - Is_admin
    - Email
    - Contraseña

El perfil de oferta ofrece las siguientes funcionalidades:

- Register en el cual se solicita la siguiente información:
    - is_daw
    - status
    - headquarter_id
    - user_id

El perfil de sede ofrece las siguientes funcionalidades:

- Register en el cual se solicita la siguiente información:
    - name
    - company_id

El perfil de empresa ofrece las siguientes funcionalidades:

- Register en el cual se solicita la siguiente información:
    - name
    - phone
    - email

3.6.- Diagramas de Análisis

3.7.- Definición de interfaces de usuario

3.7.2.- Especificación de formatos individuales de la interfaz de pantalla

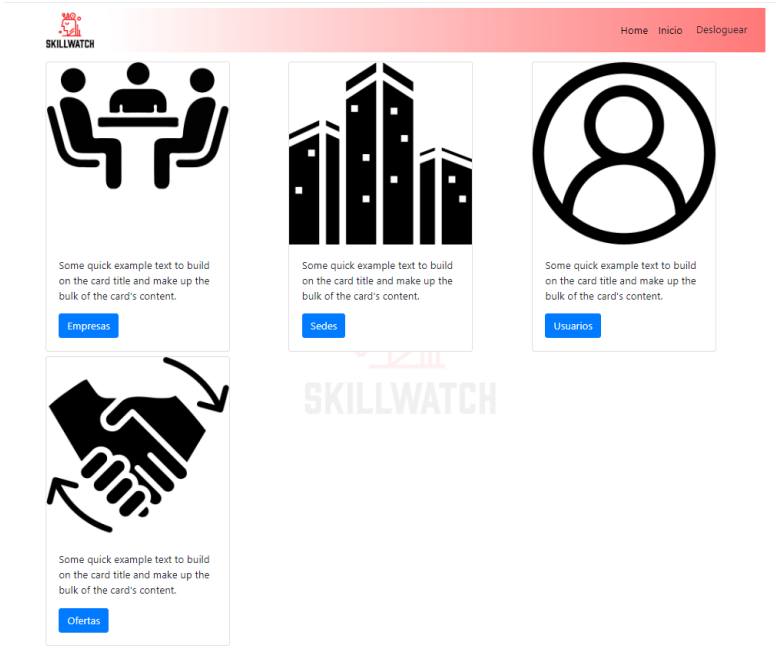

Figura 2. Captura de pantalla de la página principal de Skillwatch.

El tamaño de la página se ajusta a la resolución de la pantalla en la que se proyecta.

3.7.3.- Identificación de perfiles de usuario

Todos los usuarios tiene acceso a todas las páginas.

3.7.5.- Especificación de la navegabilidad entre pantallas

(archivo adjunto)

4.- Construcción del Sistema

(archivo adjunto)

5.- Conclusiones

En cuanto a las tareas sin finalizar, la subida del CV ha estado dando fallo hasta el final, por lo que no ha sido posible implementarla. Tampoco ha sido posible la implementación del

Crud de usuarios. Además, también se ha complicado la distinción de visibilidades y permisos dependiendo del usuario. En todos los casos por falta de tiempo y organización.

Han surgido numerosos problemas, como en el caso de la tabla offers, que ha sido dificultoso el implementar numerosas relaciones. También ha sido dificultosa la implementación del log in y el register.

Aunque el proyecto sea mejorable, he aprendido la importancia de organizarse a la hora de llevar a cabo proyectos tan extensos como este, el requerir ayuda a mis profesores y compañeros cuando me es necesaria y, en definitiva, he seguido mejorando mis habilidades como programador.