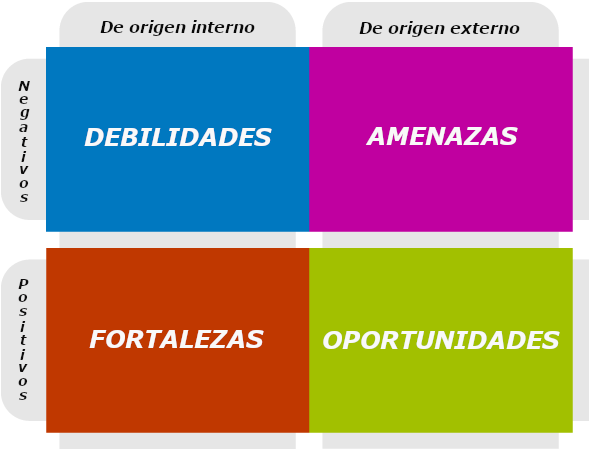
**Dafo**



### **✅ Fortalezas (Factores Internos Positivos)**

* Equipo multidisciplinar con conocimientos en desarrollo web, bases de datos, y despliegue en la nube.
* Uso de tecnologías actuales: HTML5, CSS3, JavaScript, Java (Servlets), Git, Docker, AWS.
* Enfoque realista del proyecto con aplicación práctica en orientación laboral.
* Organización mediante tableros Kanban (Jira, Trello).
* Aplicación de buenas prácticas como GitFlow, arquitectura por capas y seguridad desde el diseño.
* Presentación y documentación en inglés, útil para la empleabilidad.

### **❌ Debilidades (Factores Internos Negativos)**

* Posible falta de experiencia del equipo con herramientas cloud (AWS, RDS, EC2, Docker).
* Riesgo de mala gestión del tiempo y sobrecarga de tareas si no se distribuyen correctamente.
* Complejidad técnica elevada que puede llevar a errores si no se prueban todos los casos posibles.
* Falta de recursos reales para pruebas de rendimiento o seguridad a gran escala.

### **🌱 Oportunidades (Factores Externos Positivos)**

* Mejora directa del portafolio profesional de los alumnos.
* Alta demanda laboral de perfiles con conocimientos en tecnologías usadas en el proyecto.
* Posibilidad de reutilizar el proyecto como base para futuros desarrollos personales o empresariales.
* Fomento de soft skills (trabajo en equipo, gestión de proyectos, presentación en público).
* Desarrollo en inglés: ventaja competitiva para empleos internacionales o remotos.

### **⚠️ Amenazas (Factores Externos Negativos)**

* Cambios imprevistos en los requisitos o el entorno técnico (problemas con AWS, licencias, etc.).
* Posibles errores en el despliegue o funcionamiento en entornos reales (nube, DNS).
* Brechas de seguridad si no se aplican correctamente los principios de "Security by Design".
* Falta de conexión estable entre servicios o problemas en contenedores Docker si no se configuran bien.
* Riesgo de evaluación baja si la presentación final no comunica bien los logros del proyecto.

**SMART**

****

### **🎯 1. Desarrollo del Frontend**

**Objetivo SMART:**  
 Diseñar y desarrollar una página web responsive, intuitiva y semánticamente correcta en HTML5 y CSS que incluya roles diferenciados (cliente y trabajador), navegación entre pantallas y una sección “Trabaja con nosotros”, en inglés, antes del 10 de junio de 2025.

### **🎯 2. Presentación Final**

**Objetivo SMART:** Realizar una presentación oral final en inglés que muestre el funcionamiento completo de la web, sin uso de PowerPoint, el día pactado para la defensa del proyecto, cumpliendo con los requisitos funcionales establecidos.

### **🎯 3. Módulo de Empleo y Vacante**

**Objetivo SMART:** Publicar en la sección “Trabaja con nosotros” un anuncio para desarrollador web con requisitos mínimos y criterios de valoración claros, redactado en inglés, antes del 31 de mayo de 2025.

### **🎯 4. Gestión de la Base de Datos**

**Objetivo SMART:** Diseñar e implementar una base de datos relacional para la web, con modelo E/R, modelo lógico y físico, documentación con diccionario de datos y scripts SQL de soporte y CRUD, todo listo para conexión y funcionamiento completo antes del 5 de junio de 2025.

### **🎯 5. Arquitectura y Backend**

**Objetivo SMART:** Desarrollar una arquitectura backend basada en Servlets y DAO con conexión a base de datos desplegada en AWS (RDS), utilizando buenas prácticas y gestión de excepciones, con funcionalidades completas antes del 7 de junio de 2025.

### **🎯 6. Uso de Git y GitFlow**

**Objetivo SMART:** Gestionar el control de versiones y flujos de trabajo del proyecto mediante Git y GitFlow desde el inicio del desarrollo hasta la finalización del proyecto, con commits regulares y ramas estructuradas correctamente.

### **🎯 7. Contenerización y Despliegue**

**Objetivo SMART:** Contenerizar la aplicación (Frontend y Backend) usando Docker, subir las imágenes a Docker Hub y desplegarla en la nube (EC2 para Backend, RDS para BD y contenedor para Frontend), con nombre DNS configurado, todo completado antes del 8 de junio de 2025.

### **🎯 8. Seguridad y Protección de Datos**

**Objetivo SMART:** Aplicar principios de Seguridad by Design e implementar medidas básicas de protección de datos (cifrado, autenticación, prevención de ataques y cumplimiento con GDPR) desde el diseño inicial hasta la entrega final del proyecto.

### **🎯 9. Plan de Digitalización**

**Objetivo SMART:** Elaborar un plan de digitalización que incluya análisis DAFO, definición de objetivos SMART, alineación estratégica, plan de implantación y despliegue, completado y entregado como parte del proyecto antes del 5 de junio de 2025.

### **🎯 10. Gestión y Seguimiento del Proyecto**

**Objetivo SMART:** Utilizar una herramienta de gestión de proyectos (Trello, Jira, etc.) con un tablero compartido para el equipo donde se refleje el avance, tareas y roles, actualizado al menos 3 veces por semana durante toda la duración del proyecto.

**Alineación objetivos**

### **🎯 Objetivo General del Proyecto**

Desarrollar una página web profesional en inglés con funcionalidades completas para una empresa simulada, incluyendo una sección de empleo, seguridad, despliegue en la nube, gestión mediante base de datos y cumplimiento legal, todo con enfoque realista y profesional, orientado a la búsqueda de empleo y desarrollo empresarial.

### **🧩 Alineación de Objetivos Específicos**

| **Objetivo SMART** | **Cómo se alinea con el objetivo general** |
| --- | --- |
| **Diseño del Frontend en HTML/CSS, responsive y en inglés** | Aporta la interfaz de usuario profesional para clientes y trabajadores, clave para la imagen corporativa y la funcionalidad del sitio. |
| **Presentación final en inglés sin PowerPoint** | Desarrolla habilidades comunicativas en entorno profesional y mejora la capacidad de exposición ante clientes o empleadores reales. |
| **Sección “Trabaja con nosotros” con vacante detallada** | Facilita la simulación real de procesos de selección, alineándose con el enfoque en orientación laboral. |
| **Base de datos relacional con conexión y modelo documentado** | Garantiza la persistencia y gestión eficaz de la información, requisito técnico esencial en proyectos reales. |
| **Backend con Servlets y DAO conectados a AWS RDS** | Mejora la empleabilidad y conocimientos técnicos del equipo mediante tecnologías demandadas en el mercado. |
| **Uso de Git y GitFlow** | Fomenta el trabajo colaborativo, buenas prácticas y profesionalismo en el desarrollo del software. |
| **Despliegue en nube con contenedores y DNS** | Alinea el proyecto con tendencias actuales del sector TI, aumentando el valor del proyecto en contextos reales. |
| **Seguridad desde el diseño, protección de datos y cumplimiento legal** | Cubre aspectos legales y éticos que una empresa moderna debe cumplir, reforzando la responsabilidad profesional. |
| **Plan de digitalización (DAFO, objetivos, implantación)** | Alinea el proyecto con una visión estratégica empresarial, preparando a los alumnos para escenarios reales de transformación digital. |
| **Gestión del proyecto con tablero online** | Garantiza una buena organización, visibilidad del trabajo en equipo y control de tiempos, en línea con metodologías ágiles usadas en el sector. |

### **📌 Conclusión**

Todos los objetivos del proyecto están alineados con la visión de crear una solución web profesional, técnica y estratégica que simula un entorno real de empresa. Además, cada objetivo contribuye a la empleabilidad, profesionalización y preparación técnica de los participantes.

**Plan de implantación**

## **🧩 PLAN DE IMPLANTACIÓN – Proyecto “Staff Process”**

### **1. FASE INICIAL – Planificación del Proyecto**

#### **📌 Objetivos:**

* Desarrollar una web en inglés con funcionalidad de orientación y búsqueda de empleo.
* Desplegar en la nube una aplicación con arquitectura segura, escalable y responsive.
* Preparar una presentación final funcional, no en PowerPoint.

#### **📋 Tareas clave:**

| **Tarea** | **Herramienta/Enfoque** |
| --- | --- |
| Crear tablero de gestión de proyecto | Trello / GitHub Projects / Jira |
| Reparto de roles y responsabilidades | Scrum roles: Product Owner, Scrum Master, Developers |
| DAFO y Objetivos SMART | Documento colaborativo |
| Crear cronograma y sprints | Metodología Ágil – Scrum/Kanban |

### **2. FASE DE ANÁLISIS – Estudio previo**

#### **✅ Resultados esperados:**

* Requisitos bien definidos (funcionales y no funcionales)
* Plan de digitalización inicial (DAFO, objetivos SMART, alineación estratégica)
* Análisis de seguridad y cumplimiento legal

#### **📋 Tareas:**

* Definir requisitos funcionales (empleo, roles, CRUD, zona privada)
* Requisitos legales (GDPR, políticas de cookies, autenticación segura)
* Evaluar escenarios de resiliencia (DDoS, SQL Injection)

### **3. FASE DE DISEÑO – Frontend, Backend y Base de Datos**

#### **🎨 Frontend:**

| **Elemento** | **Descripción** |
| --- | --- |
| HTML5 Semántico | <header>, <nav>, <section>, <article>, <footer> |
| CSS + Responsive | Media queries + Flexbox/Grid |
| Navegación | JS nativo o SPA ligera con rutas internas |
| Diseño simple e intuitivo | Accesibilidad y UX (colores, tipografía, botones claros) |

#### **🧠 Backend:**

| **Elemento** | **Tecnología** |
| --- | --- |
| Java Servlets + DAO | Controlador frontal con MVC |
| CRUD de usuarios, productos, vacantes | Java + JDBC |
| Seguridad | Autenticación JWT, validación inputs |

#### **🗃️ Base de Datos:**

| **Fase** | **Herramienta** |
| --- | --- |
| Modelo E/R | dbdiagram.io, Lucidchart |
| Relacional y Físico | MySQL o PostgreSQL |
| Diccionario de datos | Documento con tablas, columnas, tipos, restricciones |
| SQL (DDL y CRUD) | Scripts de creación e inserción |
| Procedimiento almacenado adicional | Ej. enviar email semanal con nuevas vacantes |
| NoSQL funcionalidad opcional | MongoDB para logs, búsquedas rápidas, etc. |

### **4. FASE DE DESARROLLO**

#### **🔁 Control de versiones:**

* Git + GitHub
* Flujo GitFlow (develop, main, features, release, hotfix)

#### **⚙️ Desarrollo:**

* HTML/CSS responsive + navegación básica
* JavaScript Fetch con async/await para consumo de API REST
* Código modular en front y backend (carpetas organizadas)
* Acceso Role-based (cliente vs trabajador)
* Conexión a BBDD relacional (DAO)
* Gestión de errores y excepciones específicas

### **5. FASE DE INTEGRACIÓN Y PRUEBAS**

#### **🧪 Pruebas:**

| **Tipo de Prueba** | **Descripción** |
| --- | --- |
| Unitarias | Lógica backend (DAO, validaciones) |
| Funcionales | Frontend (registro, login, navegación) |
| Seguridad | SQL injection, acceso sin permiso |
| Integración | Comunicación API – DB |
| Responsive | Pruebas en móvil/tablet/escritorio |

#### **🧰 Docker y CI/CD:**

| **Elemento** | **Acción** |
| --- | --- |
| Docker HTML | Crear Dockerfile, imagen, subir a Docker Hub |
| Docker Backend Java | (Opcional) Imagen con JDK y servidor |
| Docker Compose | Orquestar front + backend + BBDD |
| CI/CD básico | GitHub Actions (lint, test, deploy) |

### **6. FASE DE DESPLIEGUE**

#### **☁️ Infraestructura:**

| **Componente** | **Plataforma** |
| --- | --- |
| Frontend | Docker en ECS / EC2 o S3 estático |
| Backend Java | EC2 o contenedor en ECS |
| Base de Datos | Amazon RDS (MySQL o PostgreSQL) |
| DNS personalizado | Route53 – dominio propio para presentación |

### **7. FASE DE PRESENTACIÓN FINAL**

* Web funcional online en inglés
* No PowerPoint: navegación directa en el navegador
* Demostración de funcionalidades (zona privada, vacantes, roles, CRUD)
* Explicación técnica breve del despliegue y la arquitectura

### **8. FASE DE DOCUMENTACIÓN**

| **Documento** | **Contenido** |
| --- | --- |
| Manual de usuario | Registro, acceso, navegación |
| Documentación técnica | Arquitectura, APIs, BD, scripts |
| Seguridad by Design | Validaciones, roles, cifrado, backups |
| Diccionario de datos | Tablas, claves, relaciones |
| Plan de contingencia | Backups, recuperación, escalabilidad |