

Proyecto de Desarrollo de Aplicaciones Web

31-05-2024

Tutor: Félix De Pablo

Institución: UNIR - FP

Título del Proyecto: Software ERP (Enterprise Resource Planning) –

Gestión de stock y pedidos de fábrica de sofás.

Autores: Alberto Saboya Ocaña, Anabella Elisabet Aceto, David Rodríguez Moral

Índice

Introducción.....	2
Módulos formativos aplicados en el trabajo.....	3
Herramientas/Lenguajes utilizados.....	7
Componentes del equipo y aportación realizada por cada estudiante.....	9
Fases del proyecto.....	10
Primera fase.....	10
Segunda fase.....	11
Tercera fase.....	13
Estudio de mercado.....	17
Modelo de datos.....	21
Diagramas UML.....	23
Diseño de las interfaces.....	24
Wireframes.....	29
Planificación del desarrollo.....	33
Explicaciones de la funcionalidad del proyecto.....	34
Conclusiones y mejoras del proyecto.....	38
Bibliografía.....	40
Anexos.....	42

Introducción

Para la elaboración del presente trabajo de fin de grado, se partió de la idea de proporcionar soluciones a un problema común en el sector de la fabricación de sofás que comprende la gestión de la fábrica respecto del control de stock de los materiales, de la productividad de los empleados y de las ventas.

Por lo expuesto, se deduce que el objetivo del proyecto se centra en mejorar la eficiencia de las empresas del mencionado sector, para lo cual se ha diseñado una aplicación web, que podrá ser implementada en un servidor interno o bien externo a la fábrica, mediante el que se podrá gestionar, en un principio, el control de los materiales, de los productos (sofás) y de la productividad de empleados.

Como se ha mencionado en el párrafo anterior, la aplicación permite gestionar el proceso de fabricación de sofás para empresas pequeñas del sector, dado que, si bien es sabido que las grandes compañías pertenecientes a este sector cuentan con un software para tal fin, las organizaciones empresariales medianas y pequeñas aún utilizan hojas de cálculo y bases de datos de ofimática para efectuar el seguimiento del stock y de la producción.

Además, cabe destacar que, en líneas generales, este tipo de aplicaciones son muy complejas de utilizar puesto que la mayoría de las herramientas disponibles en el mercado consisten en un “todo integrado”, que comprende desde el control de stock hasta la contabilidad. A los efectos de simplificar tanto la aplicación como la curva de aprendizaje, los autores se centraron en diseñar una aplicación web que permita una navegación intuitiva y una experiencia de usuario fluida, a la vez que escalable.

En suma, el objetivo se centró en crear una herramienta accesible para una amplia gama de usuarios, independientemente de su nivel de habilidad técnica, fomentando así una adopción más rápida y una mayor satisfacción con el producto final.

Módulos formativos aplicados en el trabajo

Los módulos requeridos para la elaboración del proyecto, en orden alfabético, son:

1. Bases de Datos

La utilización de esta materia se fundamenta en la necesidad de **diseñar y crear una base de datos** que se adapte a la estructura y las características inherentes a la gestión de una fábrica de muebles. Para lograrlo, se llevaron a cabo las siguientes acciones: identificación de los campos que componen las tablas del diseño lógico, análisis de las relaciones entre dichas tablas, identificación de los campos clave, aplicación de reglas de integridad, aplicación de reglas de normalización y análisis, y documentación de las restricciones que no pueden ser reflejadas en el diseño lógico.

2. Entornos de Desarrollo

Varios aspectos incluidos en esta asignatura resultaron indispensables para desarrollar el proyecto. Entre ellos, se destaca el diseño detallado de los **diagramas UML**, tanto de diagramas de uso como de clases. Asimismo, la ejecución de pruebas unitarias fue fundamental para garantizar la fiabilidad del código implementado.

Además de estos elementos clave, otros contenidos abordados en esta materia demostraron ser de gran utilidad en la implementación del proyecto. Por ejemplo, el uso de **JUnit** para la automatización de pruebas permitió validar el correcto funcionamiento de cada componente. Del mismo modo, la herramienta **Javadoc** resultó imprescindible para documentar el código de manera clara y concisa, facilitando su comprensión y mantenimiento.

3. Desarrollo Web en Entorno Cliente

Los contenidos fueron especialmente útiles para el desarrollo del frontend de la aplicación web, permitiendo al equipo crear una web dinámica e interactiva, integrando mecanismos de manejo de eventos, e implementar mecanismos de comunicación asíncrona.

Los conocimientos adquiridos sobre **manejo de eventos en JavaScript**, permitieron al equipo responder de manera adecuada a las acciones del usuario mientras que, los conceptos aprendidos sobre comunicación asíncrona como **AJAX**, facilitaron la implementación de funcionalidades como la carga dinámica de contenido y la actualización en tiempo real.

4. Desarrollo Web en Entorno Servidor.

Los aportes de esta asignatura fueron significativos en varios aspectos. Primero, permitieron el **desarrollo de una aplicación web que tiene acceso a bases de datos a través de microservicios**. Esto implica que la aplicación puede manejar grandes volúmenes de datos de manera eficiente y escalable, ya que los microservicios desacoplan diferentes partes de la aplicación y permiten que funcionen de manera independiente. Esto asegura la funcionalidad de la aplicación, lo que implica que los usuarios pueden confiar en que cumplirá con sus requisitos y operará de manera confiable.

5. Diseño de Interfaces Web.

Los contenidos aprendidos en esta materia no solo proporcionaron los conocimientos necesarios para **diseñar una interfaz de usuario**, sino que también permitieron entender y aplicar las especificaciones de diseño, los criterios de **usabilidad** y de **accesibilidad**. Estos estándares se incorporaron de manera efectiva en la aplicación web mediante la utilización de componentes visuales específicos, siguiendo guías de estilo centradas en el usuario. Esto

asegura UX(User Experience) óptima, independientemente de sus habilidades o dispositivos utilizados.

6. **Despliegue de aplicaciones web.**

La materia aportó una variedad de conocimientos y habilidades que son fundamentales en la creación de una aplicación web como la **instalación y configuración básica de servidores de aplicaciones, la instalación y configuración básica de servidores web.**

7. **Empresa e Iniciativa Emprendedora.**

Empresa e Iniciativa Emprendedora desempeñó un papel fundamental al proporcionar contenidos significativos relacionados con el **estudio del mercado**. En particular, el grupo se enfocó en la identificación de una necesidad no cubierta en el mercado y en la comprensión de los clientes potenciales. Este enfoque permitió al equipo entender las demandas del mercado y dirigir sus esfuerzos hacia la creación de un producto o que satisficiera las necesidades de manera efectiva.

8. **Inglés Técnico.**

Se trata de una materia transversal puesto que los lenguajes de programación conjuntamente con su documentación se encuentran redactados en muchas ocasiones en lengua inglesa.

9. **Lenguajes de marcas.**

Esta asignatura resultó sumamente provechosa para el proyecto, ya que brindó las herramientas necesarias para analizar de manera exhaustiva la estructura de

un documento **HTML**, identificando con precisión las distintas secciones que lo componen. Además, permitió un mayor entendimiento y reconocimiento de la funcionalidad de las principales etiquetas y atributos del lenguaje mencionado, lo que resulta fundamental para el desarrollo efectivo del proyecto y la creación de interfaces web funcionales.

10. Programación.

Los contenidos de Programación fueron fundamentales para el diseño y desarrollo del sistema de gestión, ya que permitieron a los integrantes del equipo **organizar el programa en clases**, analizando y aplicando los principios de la **programación orientada a objetos**. Asimismo, posibilitaron el uso de bases de datos orientadas a objetos, donde se analizaron sus características y se aplicaron técnicas para mantener la persistencia de la información.

11. Sistemas informáticos.

Los aportes para la construcción del software ERP se centraron en la instalación y configuración de diversos frameworks mediante el **uso de la consola de comandos**. Este proceso implicó, además, la configuración personalizada de cada uno de ellos para que se adapten adecuadamente a las necesidades específicas del proyecto. El manejo de la consola de comandos para la instalación y configuración de frameworks fue fundamental para optimizar el proceso de desarrollo del proyecto.

Herramientas/Lenguajes utilizados

- ❖ **Bootstrap:** Es un framework que facilita el diseño responsivo de los sitios web.
- ❖ **SCSS:** Es una extensión del lenguaje CSS (Cascading Style Sheets) que añade características avanzadas para mejorar la productividad y la flexibilidad en la escritura de estilos. Tiene una sintaxis similar a CSS.
- ❖ **Eclipse:** Es un entorno integrado de desarrollo para Java. Ofrece un conjunto de herramientas que facilitan la escritura de código, además de una interfaz gráfica intuitiva.
- ❖ **Figma:** Es una aplicación de diseño de interfaz de usuario que permite realizar diseño en la nube. Es una plataforma que ofrece la posibilidad de colaboración en tiempo real pudiendo visualizar los cambios inmediatamente e interactuar a través de sus chats integrados.
- ❖ **Github y Git:** Github consiste en una plataforma que permite alojar, compartir y administrar proyectos. Git es un sistema basado en el control de versiones, muy útil en los trabajos colaborativos para entornos de desarrollo dado que permite mantener el historial de cambios.
- ❖ **HTML:** Es el acrónimo de HyperText Markup Language. Es un lenguaje basado en etiquetas, reconocido mundialmente, que permite visualizar páginas web.
- ❖ **Java SE (Java Standard Edition):** Una plataforma de desarrollo de software y un entorno de ejecución para programas escritos en el lenguaje de programación Java. Permite a los desarrolladores crear una amplia variedad de aplicaciones, desde pequeños programas hasta sistemas empresariales complejos.
- ❖ **JavaDoc:** Es una herramienta que permite generar documentación, en formato HTML, a partir del código fuente de Java.
- ❖ **JavaScript:** Constituye uno de los lenguajes de “Front End” más utilizados actualmente. Es de alto nivel, interpretado y de propósito general. Con su incorporación en los frameworks tales como React, Angular y Vue.js, representa una herramienta poderosa y versátil para el desarrollo de aplicaciones web modernas.
- ❖ **Jira Software:** Se trata de una aplicación web que permite gestionar un proyecto. Cuenta con tableros de Scrum que hacen posible la división de tareas,

además de proporcionar información sobre el avance de las mismas y establecer cronogramas, entre otras herramientas.

- ❖ **JUnit:** Es un framework que permite ejecutar pruebas unitarias para comprobar que el código fuente funciona de forma adecuada.
- ❖ **MySQL Workbench:** Es una aplicación que permite desarrollar y gestionar bases de datos.
- ❖ **Postman:** Es una aplicación que permite realizar tests en una API.
- ❖ **Spring Boot:** Es un framework que brinda la posibilidad de crear aplicaciones web y microservicios con Java.
- ❖ **Visual Studio Code:** Es un editor de código que soporta varios lenguajes aunque es utilizado principalmente para el desarrollo del entorno cliente.
- ❖ **Vue.js:** Es un framework, basado en JavaScript, que se utiliza para construir interfaces de usuario interactivas basadas en componentes reutilizables. Como el framework Angular, se especializa en el desarrollo de aplicaciones de una sola página (SPA).
- ❖ **WordPress:** Es una herramienta que permite crear sitios web. Es un sistema de gestión de contenidos (CMS) gratuito y de código abierto que permite gestionar sitios web fácilmente.

Componentes del equipo y aportación realizada por cada estudiante

Alberto Saboya Ocaña

- ❖ Bases de datos.
- ❖ Backend.
- ❖ Design System.
- ❖ Landing page.
- ❖ Javadoc.
- ❖ JUnit.

Anabella Elisabet Aceto

- ❖ Bases de datos.
- ❖ Backend.
- ❖ Javadoc.
- ❖ Design Thinking.
- ❖ Redacción de la memoria del proyecto.

David Rodríguez Moral

- ❖ Bases de datos.
- ❖ Backend.
- ❖ Logo.
- ❖ Design System.
- ❖ Frontend.

Fases del proyecto

Primera fase

Durante la etapa inicial del proyecto, se llevó a cabo una investigación exhaustiva para comprender las necesidades de los usuarios y los contextos de uso. Este proceso fue fundamental para iniciar el diseño de un modelo de negocio eficiente y centrado en el usuario.

Se realizaron entrevistas presenciales con los empleados de una pequeña fábrica de sofás, lo que proporcionó información valiosa sobre sus experiencias y requerimientos específicos. Estas entrevistas permitieron obtener una perspectiva directa de los usuarios y comprender sus puntos de vista desde el aspecto operativo. La información recolectada sirvió para identificar los problemas y las áreas de mejora en los procesos actuales, así como para detectar oportunidades de innovación.

Asimismo, se utilizó la técnica de benchmarking para identificar y analizar las prácticas y productos de otras empresas similares en el mercado. Esto ayudó a contextualizar mejor las expectativas y las necesidades de los usuarios, así como a obtener información sobre las tendencias en el rubro. El benchmarking facilitó la toma de decisiones al proporcionar un marco de referencia para evaluar el desempeño y la eficacia de las estrategias propuestas. Se analizaron las funcionalidades de software E.R.P. (Enterprise Resource Planning) utilizadas por la competencia, lo que permitió establecer comparaciones y definir características diferenciadoras para nuestro proyecto.

En conjunto, la investigación documental, las entrevistas presenciales y el benchmarking fueron herramientas clave para recopilar datos relevantes, entender las necesidades de los usuarios y los contextos de uso, y establecer una base sólida para el desarrollo de un modelo de negocio centrado en el usuario. Estas actividades se enmarcaron en el proceso de Design Thinking que el equipo de trabajo inició una vez decidido el tipo de software a desarrollar. Este enfoque permitió comprender a los potenciales usuarios de la aplicación, investigar a la competencia y pensar en nuevas funcionalidades para mejorar tanto la experiencia de usuario (UX) como la interfaz de usuario (UI).

Durante el transcurso de esta fase, se procedió a la redacción del anteproyecto. En este documento, el equipo dejó constancia de las potenciales funcionalidades del software E.R.P. a desarrollar. Estas decisiones fueron tomadas de acuerdo a los resultados obtenidos en la investigación documental y el benchmarking. El anteproyecto detalló los objetivos, alcance, y las características principales del sistema, proporcionando una guía clara para las siguientes fases del desarrollo.

Otro de los aspectos resueltos fue la elección de los entornos de desarrollo y los frameworks a utilizar. Se optó por Spring Boot para el backend, Vue.js para el frontend y MySQL Workbench para la gestión de las bases de datos. Esta selección se hizo considerando la robustez, la escalabilidad y la facilidad de integración de estas tecnologías, lo que garantiza un desarrollo eficiente y una arquitectura sólida para el sistema.

En resumen, esta fase inicial del proceso permitió identificar oportunidades, desafíos y áreas de mejora, sentando las bases para la formulación de estrategias efectivas y la creación de soluciones innovadoras que respondan de manera óptima a las necesidades del mercado y los usuarios.

Puede consultarse el Design Thinking en:

https://www.figma.com/board/HtCNL2YLe8kV1kAgajvVVH/Design-Thinking_SOFA?node-id=0-1&t=XPOZcmJQgCoUwpwM-0

Segunda fase

Durante la segunda fase del proyecto, el equipo utilizó el diagrama Entidad-Relación (ER) como punto de partida para definir el modelo de bases de datos. Este diagrama permite visualizar las entidades relevantes del sistema, sus atributos y las relaciones entre ellas. Con esta información, se pudo comprender la estructura básica de la base de datos y comenzar a planificar su implementación.

Una vez que se identificaron las entidades y relaciones clave en el diagrama ER, se procedió a construir el diccionario de datos. Este diccionario sirvió como una

herramienta detallada para definir las tablas que representan las entidades en la base de datos. Cada tabla se diseñó con sus respectivos atributos, y las relaciones entre entidades se tradujeron en claves primarias y foráneas para mantener la integridad referencial.

Después de completar el diccionario de datos y asegurar que todas las entidades estuvieran correctamente representadas, el equipo avanzó con la creación de las tablas en el sistema gestor de bases de datos, en este caso, MySQL Workbench. Durante esta etapa, se definió la estructura de cada tabla de acuerdo con la información proporcionada en el diccionario de datos, asegurando la coherencia entre el diseño conceptual y la implementación física de la base de datos.

Además de crear las tablas, se especificaron las restricciones necesarias para garantizar la integridad y consistencia de los datos. Estas restricciones incluyen claves primarias, claves foráneas y restricciones de valores únicos. La definición precisa de estas restricciones fue fundamental para mantener la calidad de los datos y evitar inconsistencias.

Finalmente, una vez que se establecieron las tablas y las restricciones, se llevó a cabo la carga inicial de datos en el sistema gestor MySQL Workbench. Este paso permitió poblar la base de datos para comenzar a trabajar con el sistema y realizar pruebas de funcionalidad.

Constituida la base de datos, se procedió a construir el backend de la aplicación. Para llevar a cabo dicha tarea, como se mencionó anteriormente, se utilizó el framework Spring Boot y se emplearon herramientas como Spring Web, Spring Dev Tools, Lombok, Spring Data JPA y MySQL Server.

- ❖ Spring Web: Es un módulo de Spring que proporciona soporte para crear servicios web RESTful, manejar solicitudes HTTP y desarrollar aplicaciones web MVC. Las anotaciones `@RestController` y `@RequestMapping`, permiten definir controladores y puntos de acceso API respectivamente.
- ❖ Spring Dev Tools: Es una herramienta que ofrece características como reinicio automático del servidor cuando detecta cambios en el código, agilizando el proceso de desarrollo y prueba de la aplicación.

- ❖ Lombok: Es una biblioteca de Java que se emplea para minimizar el código repetitivo, especialmente en la creación de modelos de datos. Con anotaciones como `@Data`, `@NoArgsConstructor` y `@AllArgsConstructor`, se puede evitar escribir los getters and setters y los equals and hashCode, además de obviar la escritura de los constructores manualmente.
- ❖ Spring Data JPA: Es un subproyecto de Spring Data que facilita la implementación de repositorios basados en JPA (Java Persistence API). Permite interactuar con la base de datos de una manera más abstracta y simplificada, utilizando interfaces de repositorio y anotaciones como `@Entity`, `@Table`, `@Column` y `@Id`.
- ❖ MySQL Server: Spring Boot se integra con MySQL a través de la configuración en el archivo `application.properties`, donde se definieron las propiedades de conexión, como la URL de la base de datos, el nombre de usuario y la contraseña.

Para cada método del microservicio, se realizaron pruebas con Postman para asegurar su correcto funcionamiento. Asimismo, a medida que se fue desarrollando el backend del E.R.P., se elaboraron pruebas unitarias con JUnit, las cuales resultaron cruciales para identificar y corregir errores en etapas tempranas del desarrollo, antes de que se propagase a otras partes del sistema. Además de detectar errores, las pruebas unitarias también sirvieron para comprobar si el funcionamiento de cada método era el adecuado.

Esta segunda fase del proyecto resultó de suma importancia para establecer una base técnica sólida sobre la cual construir el sistema, asegurando que todos los componentes de backend estuvieran correctamente integrados y funcionando de manera eficiente y confiable.

Tercera fase

En la tercera y última fase del desarrollo del proyecto, el enfoque se centró en tres áreas clave: el ajuste de métodos del Backend, la continuación de las pruebas unitarias con JUnit y el desarrollo del Frontend.

En una primera instancia, se realizaron ajustes de los métodos en Spring Boot. Esta etapa resultó crucial para optimizar el rendimiento y la funcionalidad de la aplicación. Se reforzaron los métodos existentes para mejorar su eficiencia y asegurar su cumplimiento con los requisitos del proyecto. Asimismo, se implementaron nuevas funcionalidades y se corrigieron errores detectados en fases anteriores, garantizando que todos los endpoints y servicios fueran eficientes.

Simultáneamente, se continuó con la realización de las pruebas unitarias utilizando JUnit. Esta actividad fue esencial para mantener la calidad del código y para establecer que cada unidad de la aplicación funcione correctamente de manera aislada. Se escribieron nuevos casos de prueba para los métodos recientemente ajustados y se mejoraron los existentes.

En lo que respecta al desarrollo del frontend, se puso especial atención en la implementación de prácticas y principios fundamentales de experiencia de usuario (UX), con un enfoque particular en las heurísticas de usabilidad y las leyes establecidas por Jon Yablonsky. Esta parte del proyecto fue primordial para crear una interfaz de usuario atractiva y funcional.

El desarrollo del frontend se realizó utilizando Vue.js, aprovechando las ventajas de las aplicaciones de una sola página (en adelante SPA). Las SPAs ofrecen una experiencia de usuario más rápida y fluida al cargar todas las páginas desde una única solicitud al servidor y actualizar dinámicamente el contenido según sea necesario, en lugar de recargar páginas enteras. Esto mejora significativamente la rapidez y la capacidad de respuesta de la aplicación.

Se comenzó por diseñar y desarrollar los componentes básicos del software. La integración del frontend con los servicios REST (Representational State Transfer) de Spring Boot fue una tarea prioritaria, lo que permitió a los usuarios interactuar con la aplicación de manera fluida y en tiempo real. Además, se implementaron varias características de usabilidad y estética para mejorar la experiencia del usuario, como la validación de formularios, la gestión de estados y la navegación intuitiva.

Las heurísticas de usabilidad proporcionaron pautas y criterios para evaluar y mejorar la facilidad de uso de una interfaz de usuario, abordando aspectos como la accesibilidad, la

eficiencia y la satisfacción del usuario. Por otro lado, al integrar las leyes de Jon Yablonsky en el desarrollo del frontend, se buscó garantizar una experiencia de usuario óptima, donde la navegación fuese intuitiva, las acciones resultaron claras y los usuarios se sintieran comprometidos y satisfechos con la aplicación.

Por lo expuesto, se deduce que el método del “Design Thinking” constituyó una piedra angular para todo el proceso, a los fines de conseguir un diseño adecuado. Para ello, se identificaron los problemas y desafíos clave que enfrentan los usuarios al interactuar con un ERP, así como los objetivos que buscan lograr al utilizar el sistema. Esto ayudó a establecer una dirección clara para el diseño del frontend, enfocándose en resolver problemas específicos y satisfacer las necesidades del usuario.

Asimismo, dentro del enfoque del Design Thinking, se consideró la accesibilidad como un componente fundamental del diseño. Se reconoció la importancia de garantizar que el sistema fuera accesible para todos los usuarios, independientemente de sus capacidades físicas o cognitivas.

Luego, se generaron múltiples ideas y soluciones para abordar los problemas identificados utilizando técnicas de brainstorming. La lluvia de ideas no sólo permitió la colaboración, también resultó ser un factor clave para generar ideas, construir sobre las ideas de los demás y pensar de manera más abierta.

El desarrollo del Frontend comenzó con la identificación de los componentes básicos necesarios para la aplicación. En Vue.js, los componentes son unidades reutilizables de código que pueden manejar su propio estado y lógica.

Uno de los primeros componentes desarrollados fue el botón, un elemento fundamental en cualquier aplicación. Se creó un componente de botón genérico que se podría reutilizar en toda la aplicación. Este componente incluía propiedades personalizables como el texto del botón, el tipo (add, edit, delete, cancel, confirm), y eventos como “@click”.

La creación de vistas fue el siguiente paso. Vue Router se utilizó para gestionar la navegación entre las vistas, permitiendo una experiencia de usuario sin interrupciones típica de las aplicaciones SPA.

Las vistas principales incluyen:

- ❖ Vista de inicio: Presenta una visión general de la aplicación, con enlaces a las secciones más importantes. En esta vista se muestra, en gráficos, el estado de las tareas por cada departamento de producción, en forma de gráfico de barras.
- ❖ Vista de materiales: Muestra una lista de materiales, permitiendo a los usuarios ver detalles y realizar acciones a través de los botones “nuevo”, “editar” y “eliminar”.
- ❖ Vista de proveedores: Brinda a los usuarios la posibilidad revisar y gestionar los proveedores.
- ❖ Vista de sofás: Muestra una lista de los detalles de los pedidos, a la vez que permite acciones como editar, agregar y eliminar cada producto. A través del botón materiales se puede acceder a la lista de materiales necesarios para la producción del sofá.
- ❖ Vista de pedidos: Consta de una lista de todos los pedidos con las respectivas acciones asociadas a un CRUD. Incluye una columna en la que se muestra el estado general del pedido.
- ❖ Vista de clientes: Brinda a los usuarios la posibilidad de revisar y gestionar los clientes.
- ❖ Vista de empleados: Presenta una lista de todos los empleados. Permite actualizar los datos de un empleado, consultar sus datos y gestionar su estado.
- ❖ Formularios de alta y edición con campos para input y dos botones de acción: “confirmar” y “cancelar”.

Por último, se creó la “landing page” con WordPress, manteniendo el diseño del wireframe. Se creó la web en un servidor local con la aplicación Local. Se utilizaron plugins como Elementor, All-in-One WP Migration, WP Menu Icons y Font Awesome. El tema utilizado para la web fue Kadence. Cuando se terminó la web, a continuación, se adquirió un hosting y un dominio en Hostinger. Por último se hizo la migración de la web desde el servidor local hasta el servidor web.

URL para la landing page: <https://proges.cloud/>

Estudio de mercado

La idea de diseñar un software de gestión de fabricación de sofás surge a partir de las necesidades no cubiertas para las pequeñas empresas que no pueden afrontar los costos que demandan los sistemas gestores y la complejidad de las aplicaciones actuales para el usuario final.

Asimismo, se detectó que las empresas oferentes de este tipo de aplicaciones abarcan diferentes rubros. Por lo tanto, la falta de un software específico para las fábricas que se dedican a la producción de sofás, constituyó en un factor clave a la hora de decidir la puesta en marcha del proyecto.

Según un artículo publicado en [Business Research Insights](#) el interés por los sistemas que gestionan pedidos se encuentra en crecimiento aunque, entre las restricciones en su demanda se encuentra la complejidad que implica la implementación de estos sistemas.

En un artículo titulado “de ERP para 2023 y más allá” publicado por [SAP](#), las empresas que se han adaptado rápidamente en el período de la pandemia, lo han hecho utilizando sistemas ERP modernos, esenciales para la innovación y flexibilidad. Entre las principales tendencias de ERP para 2023 y más allá destacan: la adopción creciente del ERP en la nube, experiencias móviles integradas en múltiples dispositivos, y la incorporación de IA e IoT.

Por último, en un tercer artículo publicado por [Mordor Intelligence](#), se menciona que el mercado de planificación de recursos empresariales (ERP) se estima en \$65,25 mil millones en 2024, con proyecciones de alcanzar \$103,95 mil millones en 2029, creciendo a una tasa anual del 9,76%.

En resumen, la falta de un software específico para las fábricas de sofás fue un factor determinante en la decisión de emprender este proyecto. Además, los sistemas ERP modernos, con tendencias como la adopción en la nube, experiencias móviles integradas y la incorporación de IA e IoT, están en aumento según los expertos. El mercado de planificación de recursos empresariales se proyecta crecer significativamente en los próximos años .

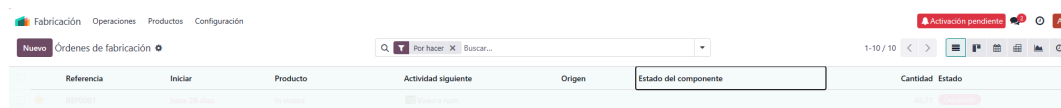
1. [ODOO](#)

Ofrece módulos de aplicaciones para las empresas. El usuario debe registrarse y, una vez logueado, podrá instalar en su equipo el módulo correspondiente.

Ofrece los siguientes módulos:

FINANZAS Contabilidad Facturación Gastos Hoja de cálculo (BI) Documentos Firma electrónica	VENTAS CRM Ventas Punto de venta - Tiendas Punto de venta - Restaurante Suscripciones Alquiler	SITIOS WEB Creador de sitios web Comercio electrónico Blog Foro Chat en vivo eLearning	INVENTARIO Y MRP Inventario Fabricación PLM Compra Mantenimiento Calidad
RECURSOS HUMANOS Empleados Reclutamiento Tiempo personal Evaluación Referencias Flota	MARKETING Marketing social Marketing por correo electrónico Marketing por SMS Eventos Automatización de marketing Encuestas	SERVICIOS Proyecto Hojas de asistencia Servicio externo Servicio de asistencia Planificación Citas	PRODUCTIVIDAD Conversaciones Aprobaciones IoT VoIP Información Nueva aplicación WhatsApp

La vista de las órdenes de fabricación es la siguiente:



La vista para imprimir una nueva orden de fabricación es la siguiente:

Confirmar

Imprimir etiquetas

Borrar

Confirmado

Hecho

☆ New

Producto

Producto a construir...

Fecha prevista ?

29/04/2024 12:02:25

Cantidad

1,00

A producir

Responsable

Mercedes Calla

Lista de materiales ?

Componentes

Varios

Producto

A consumir

Agregar línea

Catálogo

En cuanto a los costos, ofrecen una aplicación gratuita y luego, suscripciones con tarifas que se abonan mensualmente por usuario.

Otros datos importantes:

- ❖ El proceso de registro en Odoo requiere información básica del usuario, como nombre, dirección de correo electrónico y contraseña. Algunas empresas

también pueden requerir información adicional para personalizar la experiencia del usuario.

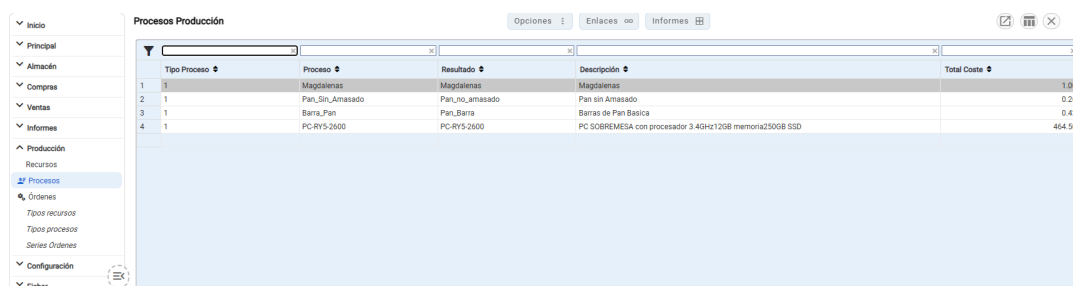
- ❖ Una vez que el usuario se registra en Odoo, puede acceder a su cuenta desde cualquier dispositivo con conexión a internet.
- ❖ Después de iniciar sesión, el usuario puede explorar el mercado de aplicaciones de Odoo y seleccionar los módulos que mejor se adapten a las necesidades de su empresa.
- ❖ Para instalar un módulo en su equipo, el usuario puede hacer clic en el botón de instalación correspondiente en la página del módulo. Dependiendo de la configuración de la empresa y del tipo de implementación de Odoo (en la nube o local), la instalación puede variar.
- ❖ Una vez instalado, el módulo estará disponible para su uso inmediato, y el usuario puede comenzar a aprovechar sus características y funcionalidades para mejorar la eficiencia y productividad de su negocio.

2. MyGestion

Es un software de gestión en la nube. En la página se brinda información sobre los costos de cada uno de los módulos. Ofrecen un período de quince días de prueba gratuita para lo cual se debe realizar un registro. Recibido el email, se activa la cuenta y el período gratuito de prueba.

El software se encuentra estructurado en módulos, los cuales se pueden añadir. También, y ofrece la opción de crear datos ficticios.

Al ingresar, se visualiza una barra de navegación lateral para la selección de la vista.



The screenshot shows the 'Procesos Producción' (Production Processes) section of the MyGestion application. On the left is a sidebar menu with options like Inicio, Principal, Almacén, Compras, Ventas, Informes, Producción, Recursos, Órdenes, Configuración, and Fichas. The main area displays a table with columns: Tipo Proceso, Proceso, Resultado, Descripción, and Total Costo. The table contains four rows of data related to 'Magisemas' and 'Pan' products.

	Tipo Proceso	Proceso	Resultado	Descripción	Total Costo
1	3	Magisemas	Magisemas	Magisemas	1.08
2	1	Pan_Sin_Amasado	Pan_no_Amasado	Pan sin Amasado	0.25
3	1	Barra_Pan	Pan_Barra	Barra de Pan Basica	0.42
4	1	PC-RY5-2600	PC-RY5-2600	PC SOBREMESA con procesador 3.4GHz 12GB memoria 250GB SSD	464.50

En el menú superior, se muestran las opciones para cada módulo.

Procesos Producción

Tipo Proceso	Proceso	Resultado	Total Coste
1	Magdalenas	Magdalenas	1.00
2	Pan_Sin_Amasado	Pan_no_amasado	0.26
3	Barra_Pan	Barra de Pan Basica	0.42
4	PC-RYS-2600	PC-RYS-2600	464.50

Opciones:

- Generar Orden Producción
- Duplicar Proceso
- Actualizar coste de los artículos en los procesos seleccionados

Artículos

Código	Código de Barras	Descripción	Venta	Con IVA	Stock	Pend.Recibir	Pend.Servir
15031-2	8431707067614	Lotus Classic 15031-2	42.90	51.91	0.00	20.00	20.00
15150-A	8431707067621	Lotus Classic 15150-A	58.75	71.09	0.00	0.00	0.00
15171-1	8431707067638	Lotus Classic 15171-1	58.75	71.09	0.00	30.00	30.00
15351-6	8431707067645	Lotus Titanium 15351-6	200.00	242.00	0.00	8.00	8.00
15351-C	8431707067652	Lotus Titanium 15351-C	106.25	128.56	0.00	0.00	0.00
15351-D	8431707067669	Lotus Titanium 15351-D	106.25	128.56	0.00	15.00	15.00
860_EVO	8431707067676	Samsung 860 EVO Basic SSD 250	46.27	55.99	6.00	2.00	0.00
A142	8431707067683	Proyector Casio XJ-A142	100.00	121.00	0.00	0.00	0.00
A257	8431707067690	Proyector Casio XJ-A257	162.50	196.62	0.00	0.00	0.00

Opciones:

- Regularizaciones de Stock
- Historial de Movimientos
- Números de Serie
- Lotes y Caducidad
- Precios
 - Duplicar Artículo
 - Composición de Kits
 - Asignar %Margen a Artículos Seleccionados
 - Borrado de Artículo Y TODOS sus datos
 - Generar recurso en producción
 - Generar Recursos en producción de los seleccionados

Otros datos importantes:

- ❖ El software ofrece funcionalidades como gestión de clientes, facturación, inventario, recursos humanos, entre otros.
- ❖ La página web también proporciona soporte técnico y tutoriales para el uso eficiente del software.
- ❖ Durante el período de prueba gratuita, los usuarios tienen acceso completo a todas las características y funcionalidades del software.
- ❖ El registro para la prueba gratuita requiere información básica del usuario, como nombre, dirección de correo electrónico y contraseña.
- ❖ Después de completar el registro, los usuarios reciben un correo electrónico de confirmación con instrucciones para activar su cuenta.
- ❖ Una vez activada la cuenta, los usuarios pueden comenzar a utilizar el software de inmediato y explorar sus capacidades.
- ❖ La opción de crear datos ficticios permite a los usuarios simular situaciones y escenarios reales para probar la funcionalidad del software sin afectar datos sensibles.
- ❖ La barra de navegación lateral proporciona acceso rápido a las diferentes secciones y módulos del software, facilitando la navegación y la búsqueda de funciones específicas.

Modelo de datos

Para el diseño de este proyecto, se optó por la utilización de una base de datos relacional. Para comenzar con su diseño, primeramente se procedió a la elaboración del diagrama entidad-relación. Este paso resultó de fundamental importancia dado que, a través de este, se representaron los objetos y sus relaciones para identificar las interacciones entre esas entidades.

El diagrama puede consultarse en:

<https://www.figma.com/board/o8nmz9bkDzDhp1JYx6E3jD/Diagrama-Entidad%2FRelacion?node-id=0-1&t=DmphavUv2AuCZift-0>

Luego, se procedió a la construcción del diccionario de datos, puesto que este proporciona una lista completa de todas las entidades y sus atributos. De esta manera se estableció una convención de nomenclatura coherente.

Diccionario de datos

CLIENTES (PK(id_cliente)), nombre, apellidos, email, teléfono)

EMPLEADOS (PK(id_empleado), nombre, apellidos, (FK(id_depto)), (FK(id_perfil)), fecha_ingreso, fecha_baja, estado, salario)

PEDIDOS (PK(id_pedido), (FK(id_cliente)), (FK(vendedor)), fecha)

DETALLE_PEDIDO(PK(id_deped), FK(id_pedido), (FK(id_sofa)), cantidad, fecha, estado, plazas, dens_cojín)

DEPARTAMENTOS ((PK(id_depto)), nombre)

TAREAS ((PK(id_tareas)), (FK(id_empleado)), (FK(id_depto)), (FK(id_estado)), FK(id_deped), fecha)

PERFILES((PK(id_perfil)), nombre)

ESTADOS(PK(id_estado)), nombre)

MATERIALES(PK(id_material)), nombre, descripcion, cantidad, FK(id_proveedor), ref_material_prov, categoria, unidad_medida)

SOFAS(PK(id_sofa)), FK(id_material), nombre, descripcion, patas, medida_cojin, precio)

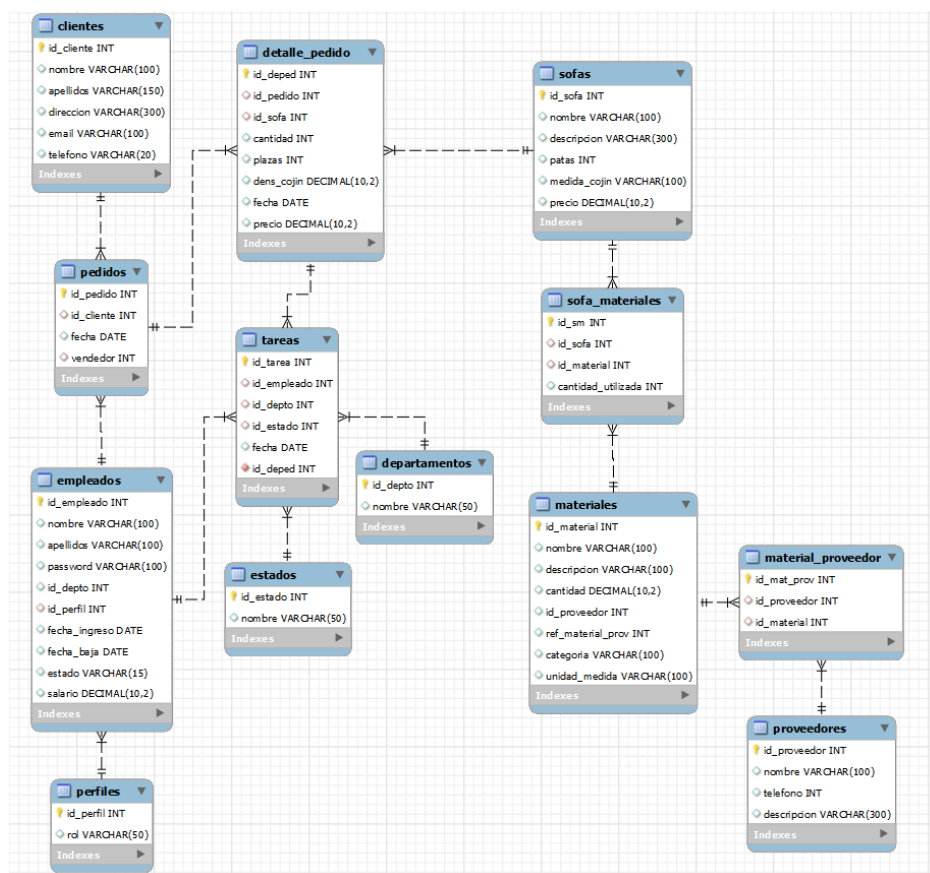
PROVEEDORES(PK(id_proveedor)), nombre, telefono, descripcion)

Durante la normalización se detectó que era necesario crear dos tablas intermedias para garantizar la tercera forma normal, a saber:

SOFA_MATERIALES(PK(idSM), FK(idSofa), FK(idMaterial))

MATERIAL_PROVEEDOR(PK(idMP), FK(idProveedor), FK(idMaterial))

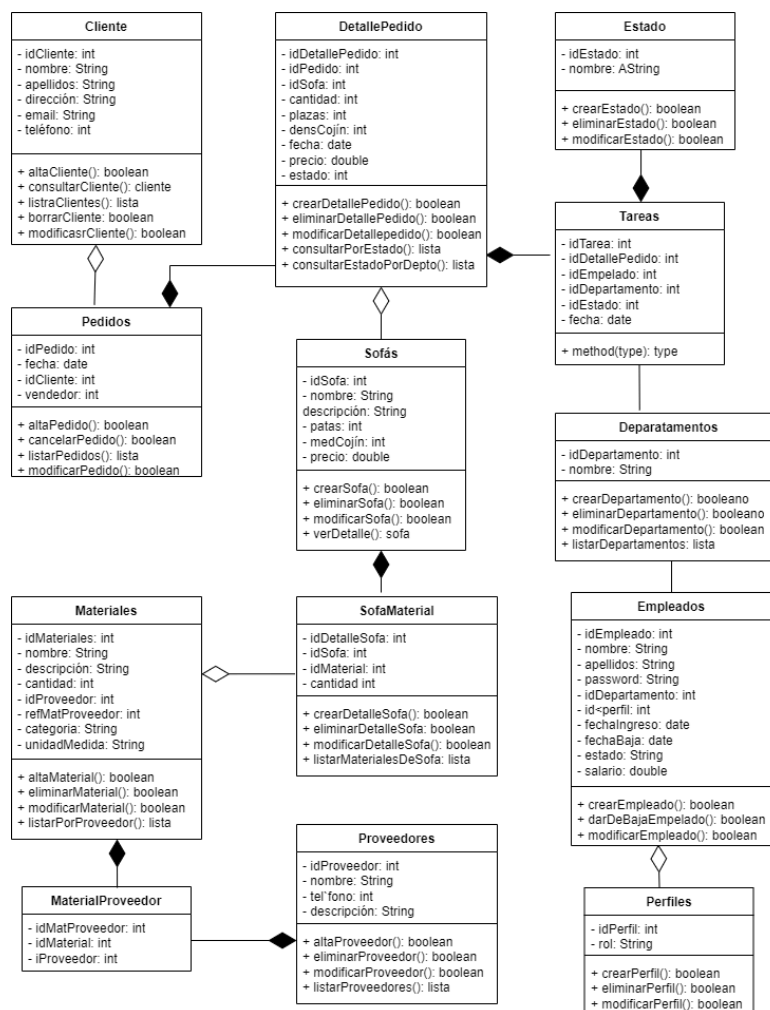
La estructura de la base de datos resultó ser:



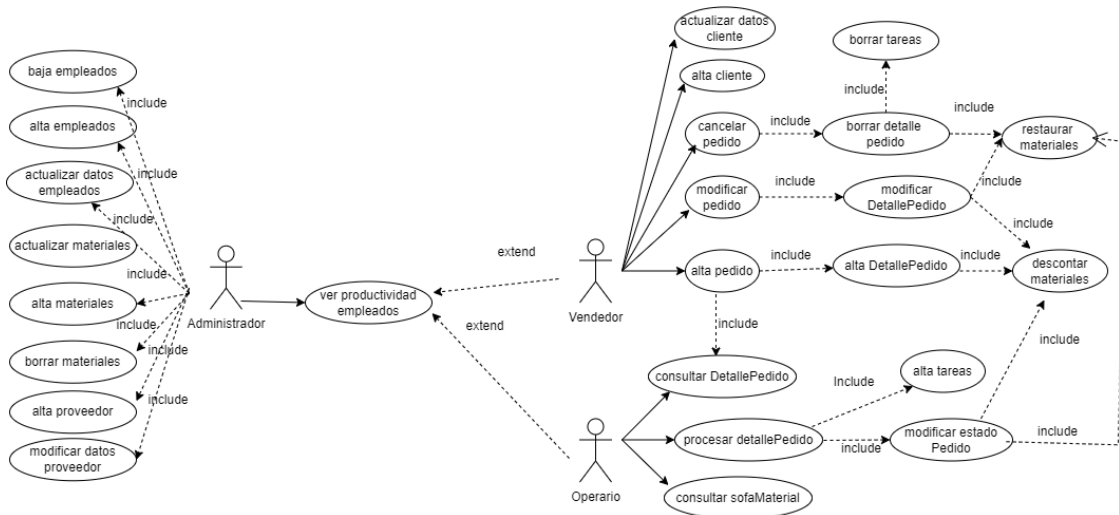
Diagramas UML

El diagrama de clases y el diccionario de datos están estrechamente relacionados en materia de diseño de bases de datos. El diagrama de clases, por su parte, muestra las clases del sistema y las relaciones entre ellas de manera pictórica mientras que, el diccionario de datos, proporciona el detalle de las entidades, los atributos y las relaciones en forma de texto.

La representación de las entidades y sus atributos en un diagrama de clases resultó útil para identificar los problemas en el diseño de la base de datos. Detectados los inconvenientes, se revisaron las relaciones entre las clases para identificar las inconsistencias, además de verificar los atributos de cada clase para asegurar que estuvieran correctamente definidos y asociados con las entidades adecuadas. Por último, se procedió a mejorar la estructura reflejando todas las correcciones y mejoras realizadas en el diagrama de clases en el diccionario de datos.



Los diagramas de casos de uso permiten modelar conceptos como los procesos de negocio y las funciones del sistema a nivel general. Es por esto que, para detectar problemas respecto de las funciones y de los objetivos más importantes del software.



Diseño de las interfaces

Es ampliamente reconocido que los modelos mentales de los usuarios desempeñan un papel crucial en la configuración y el éxito de una interfaz web. Estos modelos, que abarcan percepciones, expectativas, habilidades y emociones, influyen significativamente en cómo los usuarios interactúan con un sitio web. Por ende, al diseñar una interfaz web, es imperativo que el modelo de interacción esté firmemente arraigado en la experiencia y la perspectiva del usuario, priorizando la usabilidad y la coherencia.

Para abordar esta necesidad, se desarrolló un sistema de diseño que tomó en consideración los diversos aspectos de los modelos mentales de los usuarios. Este enfoque se centró en comprender y responder a sus requisitos, capacidades y expectativas. Al integrar estos elementos en el proceso de diseño, se buscó crear una experiencia coherente y consistente que resonara con los principios fundamentales del diseño y los modelos mentales de los usuarios.

La accesibilidad y la legibilidad constituyeron las bases sobre las cuales se erigió todo el diseño de la interfaz de usuario. En este sentido, se adoptaron una serie de elementos clave dentro del sistema de diseño para asegurar una experiencia de usuario óptima. Una

interfaz complicada, llena de características, que no respete el principio de “Affordance¹”, puede generar una experiencia de usuario (UX) negativa.

El logotipo, por ejemplo, se ubicó estratégicamente en la esquina superior izquierda siguiendo los principios fundamentales de diseño de logos. Estos principios incluyen la simplicidad, la escalabilidad, la coherencia tipográfica y la claridad visual, entre otros aspectos. Un logotipo bien diseñado facilita a los usuarios identificar y recordar el sitio web, mejorando así la experiencia de usuario desde el primer contacto.



El logo contiene el nombre de la organización (PROGES) y contiene la primera sílaba de las palabras “producción” y “gestión”.

Además, se establecieron pautas específicas para los botones, que se presentan en recuadros de colores distintivos con efectos de "hover" para indicar interactividad al usuario. Los botones son elementos cruciales de la interfaz, ya que guían la navegación y la interacción del usuario. Cada botón fue diseñado para ser claramente visible y accesible, utilizando colores y efectos visuales que resaltan su función y estado.

La paleta de colores y la tipografía fueron cuidadosamente seleccionadas para garantizar una experiencia visualmente agradable y accesible. Se aplicó la regla del "60-30-10" en la elección de colores, donde el 60% corresponde a un color dominante, el 30% a un color secundario y el 10% a un color de acento. Esta regla ayuda a crear una jerarquía

¹ "Affordance" es un término psicológico introducido por D. Norman que se refiere al aspecto del diseño de un objeto que sugiere cómo debe usarse dicho objeto.

visual clara y atractiva. Además, se aseguró un contraste de accesibilidad AAA y se eligieron colores poco saturados para promover la legibilidad y la armonía visual.

En cuanto a la tipografía, se optó por un enfoque minimalista, utilizando estilos derivados de las fuentes sans-serif como “Arial Rounded” e “Inter” para mantener la cohesión y la legibilidad en todo el diseño. Las fuentes sans-serif son conocidas por su claridad y simplicidad, lo que facilita la lectura en pantallas de diferentes tamaños y resoluciones.



Se implementaron colores específicos para los botones de feedback, como información, éxito, advertencia y error, con el fin de comunicar de manera efectiva diferentes estados y acciones al usuario. Estos colores ayudan a los usuarios a comprender rápidamente el estado de una acción o la importancia de un mensaje, mejorando la eficiencia y efectividad de la comunicación en la interfaz.

Para la creación de los botones, tanto de la aplicación como de la "landing page", se prestó especial atención a que estos fueran claros y específicos al comunicar la tarea, llevando la etiqueta apropiada (en algunos casos acompañada de icono), y ejecutando una acción predecible para el usuario.

El botón "eliminar" con tono rojo tiene su razón de ser en su naturaleza llamativa y vibrante. Este color captura la atención del usuario de manera inmediata, lo cual resulta especialmente útil dado que el acto de eliminación es una acción importante e irreversible. Como estrategia de diseño, esta tonalidad combina atención visual, asociaciones culturales y efectos psicológicos para comunicar claramente la importancia y urgencia de la acción.

En cuanto a los botones de "confirmar" o "agregar", se optó por el color verde, puesto que combina asociaciones positivas, efectos psicológicos y legibilidad para mejorar la experiencia del usuario. El color verde, asociado con el éxito y la confirmación, ayuda a los usuarios a sentir seguridad y claridad al realizar acciones en la interfaz.



Color de base



Colores de marca



Colores de contraste



Colores para acciones

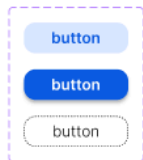


KIT DE COMPONENTES

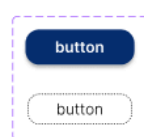
Botones/Tarjetas/Formularios



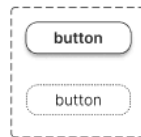
Botón vista app



Botón probar gratis



Botón solicitar demo



Tarjeta



La adición de iconos en cada botón se utilizó con la intención de reforzar el significado de la acción asociada. Los iconos, al ser elementos visuales universales, facilitan la comprensión rápida y efectiva de las funciones de los botones, especialmente para usuarios con diferentes niveles de alfabetización o fluidez en el idioma.

Finalmente, la iconografía general se integró de manera discreta, utilizando iconos simples que complementan la estética general de la página sin distraer ni interrumpir la experiencia del usuario. Estos iconos, diseñados para ser intuitivos y fáciles de reconocer, contribuyen a una navegación más fluida y eficiente.

ICONOGRAFÍA

Iconos



Redes sociales



Contacto



Misceláneas



En resumen, cada aspecto del sistema de diseño se esquematiza teniendo en cuenta los modelos mentales de los usuarios para ofrecer una UX fluida y satisfactoria. Este

enfoque asegura que la interfaz no solo sea visualmente atractiva, sino también funcional y accesible, alineándose con las necesidades y expectativas de los usuarios.

Cabe destacar que cada uno de los elementos mencionados ha sido incorporado a la librería de componentes, siguiendo la metodología del "Atomic Design" ideada por Brad Frost. Este enfoque fusiona la disciplina del diseño y del desarrollo bajo el concepto de diseñar por componentes, lo que permite optimizar los elementos de diseño en el proceso de desarrollo de una página web. El "Atomic Design" facilita la creación de interfaces modulares y escalables, asegurando la coherencia y reutilización de los componentes a lo largo del proyecto.

Wireframes

Los wireframes que se muestran a continuación corresponden a las vistas de la aplicación.

Vista de inicio



Aspecto de la vistas de materiales, proveedores, clientes y empleados



Materiales Proveedores Pedidos Sofás Clientes Empleados Departamentos


Lista de Materiales

+ Nuevo

Buscar...

	<div>Editar</div> <div>Eliminar</div>
	<div>Editar</div> <div>Eliminar</div>
	<div>Editar</div> <div>Eliminar</div>
	<div>Editar</div> <div>Eliminar</div>

Formulario para añadir o editar



Materiales Proveedores Pedidos Clientes Sofás Empleados Departamentos

Referencia1

Opciones

Referencia3

Fecha

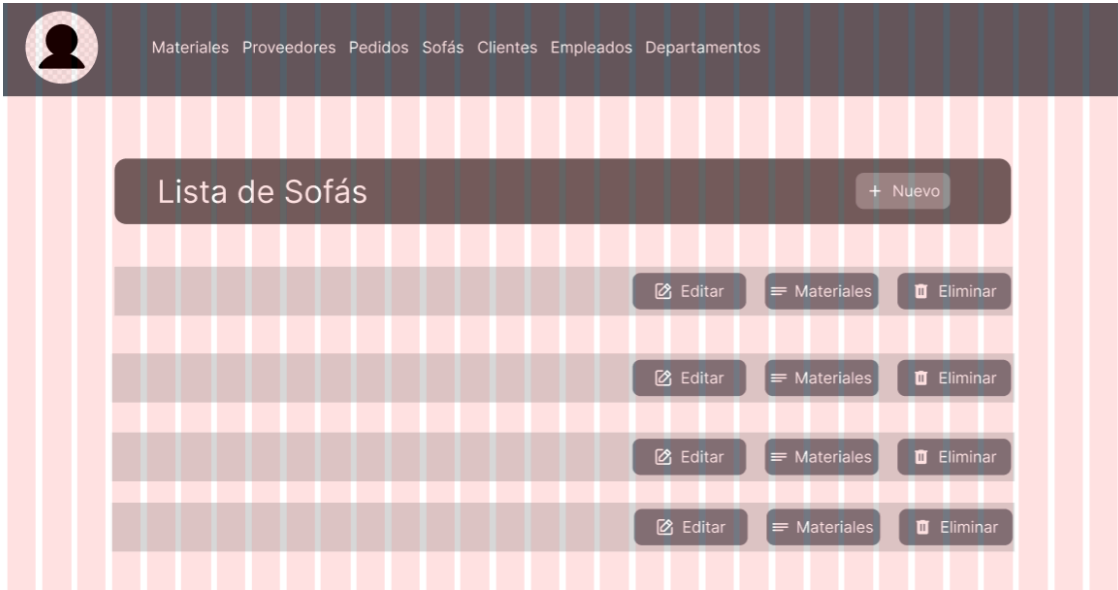
dd/mm/aaa

Referencia2

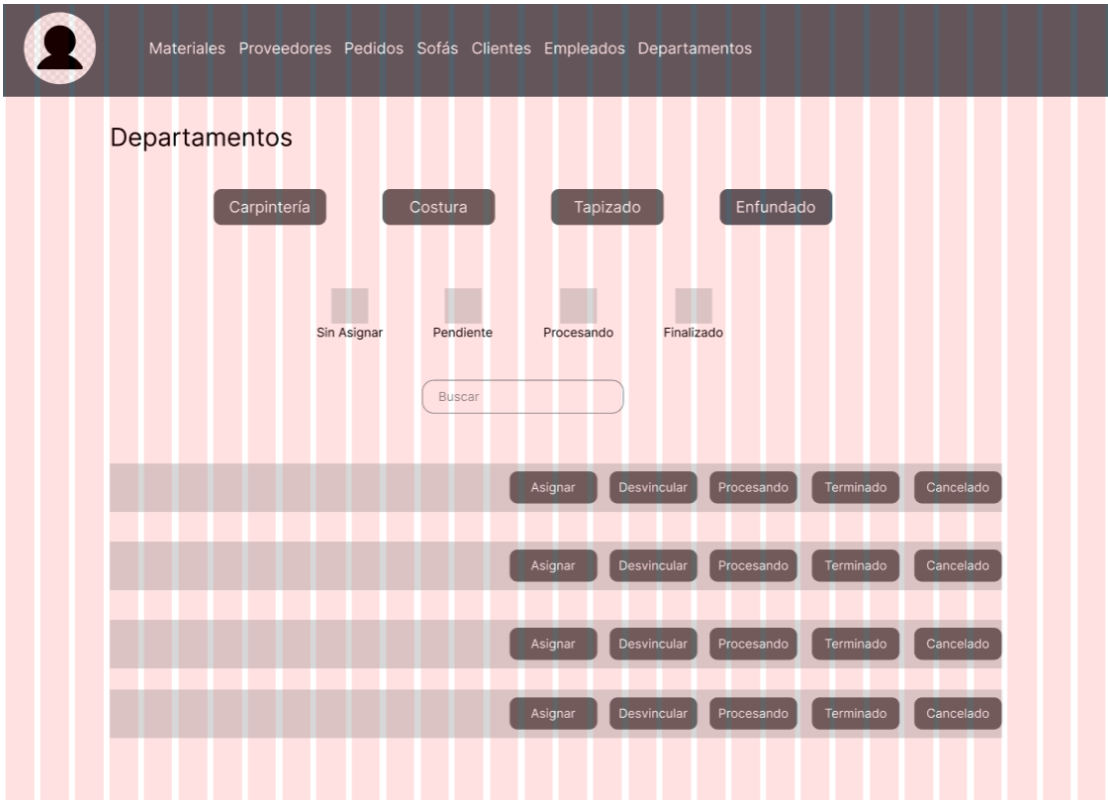
× Cancelar

✓ Confirmar

Vista de la lista de sofás



Vista de los departamentos





Planificación del desarrollo

Para la planificación del desarrollo del proyecto se utilizó el software Jira. La decisión de optar por el uso de esta plataforma radicó en su potencialidad y su funcionalidad.

La plataforma cuenta con una vista de cronograma que permite planificar el trabajo, hacer un seguimiento del progreso y asignar las dependencias dentro del proyecto.



Además, cuenta con una vista que muestra una lista de las tareas y el estado en el cual se encuentran las mismas.

Resumen

Tablero

Lista

Calendario

Cronograma

Aprobaciones

Formularios

Páginas

Adjuntos

Incidencias

Informes

Atajos de teclado

Buscar en la lista

AA ALB

Compartir

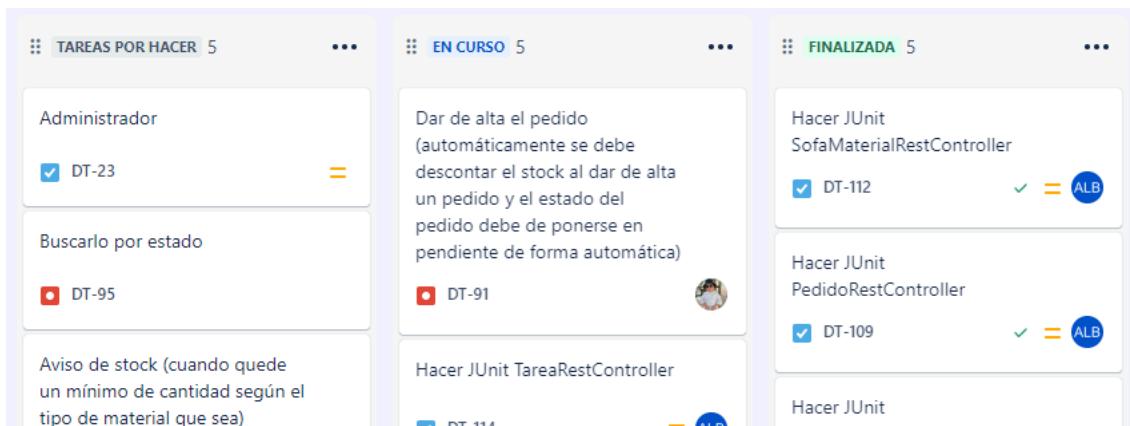
Filtro

Grupo

...

<input type="checkbox"/>	Tipo	# Clave	Resumen	Estado	Categoría	@ Perso
<input type="checkbox"/>		DT-100	Se da de alta la tarea de forma automática cuando el emplea...	FINALIZADA		AA Anz
<input type="checkbox"/>		DT-107	HacerJUnit EstadoRestController	FINALIZADA		ALB Alb
<input type="checkbox"/>		DT-92	Cancelar pedido (si un pedido es cancelado se deben de su...	FINALIZADA		AA Anz
<input type="checkbox"/>		DT-77	Hacer JavaDoc	FINALIZADA		AA Anz
<input type="checkbox"/>		DT-103	Buscar el detalle de un pedido, pasándole el id del pedido	FINALIZADA		AA Anz
<input type="checkbox"/>		DT-96	Buscar tareas que ha hecho un empleado	FINALIZADA		AA Anz

En Jira, el tablero muestra, en columnas, el flujo de trabajo del equipo. Este tablero proporciona a los usuarios una vista compartida de todas las tareas que no se han iniciado aún, las que están en curso y las que ya se han completado.



En la página principal, se puede visualizar un resumen del estado de las tareas realizadas por el equipo y las que aún se encuentran pendientes de realizar o en proceso de ejecución.



Explicaciones de la funcionalidad del proyecto

El software que se desarrolló ofrece una amplia gama de funcionalidades diseñadas para mejorar la eficiencia y la gestión en el sector de la fabricación de sofás. Aquí se detallan las principales funciones que incluye:

1. **Gestión de materiales y stock:** Permite llevar un control detallado del stock de materiales necesarios para la fabricación de sofás, incluyendo la posibilidad de realizar consultas de detalle, añadir nuevos materiales, modificar datos de materiales existentes y recibir alertas para reabastecimiento.

Cuando se intenta crear el detalle de un pedido, los materiales requeridos para la fabricación del producto son descontados del stock. Si el material es

insuficiente, se emite una alerta indicando qué material está en faltante y se cancela la inserción del detalle.

2. Gestión de proveedores: Facilita el control de los proveedores de materiales y de los elementos necesarios para la fabricación de sofás, con opciones para añadir nuevos proveedores, consultar datos de los proveedores, modificar sus datos y eliminarlos.
3. Catálogo de modelos de sofás: Incluye un catálogo completo de modelos de sofás disponibles, con detalles sobre la cantidad de materiales requeridos para la manufactura de cada modelo y su coste asociado. Los operarios podrán acceder a estas plantillas al momento de comenzar a procesar el pedido. El administrador podrá agregar, modificar o eliminar un modelo de sofá según necesidad.
4. Control y seguimiento de pedidos: Permite gestionar y realizar un seguimiento detallado de todos los pedidos, desde su creación hasta su finalización. Se podrán añadir nuevos pedidos, consultar detalles, modificar el estado del pedido y eliminar pedidos según sea necesario.
5. Control y gestión del detalle de un pedido: Permite gestionar el detalle de un pedido, que se encuentra asociado a un pedido.
 - 5.1. El administrador y los empleados pueden acceder a la lista de todos los detalles de pedido.
 - 5.2. Al crear un detalle de pedido, se restan los materiales asociados al sofá que se confeccionará y se dan de alta las tareas en cada uno de los departamentos encargados de su producción.
 - 5.3. Se crea un detalle de pedido por cada modelo de sofá y por cada unidad solicitada.
 - 5.4. Puede consultarse el estado general de un pedido o bien, el estado de producción en el que se encuentra en cada departamento. En el caso de querer consultarse el estado general de un pedido, se obtiene el estado de las tareas en cada departamento de cada uno de sus detalles asociados y se emite una generalización conforme a las siguientes reglas:

- ❖ Todos las tareas de los departamentos, de todos los detalles asociados a un pedido tienen el estado “sin asignar”: se devuelve como estado general “sin Asignar”.
- ❖ Todos las tareas de los departamentos, de todos los detalles asociados a un pedido tienen el estado “finalizado”: se devuelve como estado general “finalizado”.
- ❖ Todos las tareas de los departamentos, de todos los detalles asociados a un pedido tienen el estado “pendiente”: se devuelve como estado general “pendiente”.
- ❖ Todos las tareas de los departamentos, de todos los detalles asociados a un pedido tienen el estado “procesando”: se devuelve como estado general “procesando”.
- ❖ Algunas tareas de los departamentos, de algunos de los detalles asociados a un pedido tienen el estado “sin asignar” y otros tienen el estado “pendiente”: se devuelve como estado general “pendiente”.
- ❖ Algunas tareas de los departamentos, de algunos de los detalles asociados a un pedido tienen el estado “sin asignar” y otros tienen el estado “procesando”: se devuelve como estado general “procesando”.
- ❖ Algunas tareas de los departamentos, de algunos de los detalles asociados a un pedido tienen el estado “pendiente” y otros tienen el estado “procesando”: se devuelve como estado general “procesando”.

5.5. La actualización de un detalle de pedido implica la restauración (descuento) de los materiales requeridos para elaborar el sofá en cuestión. Por ejemplo, un cliente decide que en vez de querer cuatro unidades de un tipo de sofá, quiere dos unidades de un producto y dos unidades de otro producto. En este caso, se restauran en el stock los materiales correspondientes a las dos unidades devueltas y se descuentan los materiales que se corresponden al nuevo producto seleccionado.

6. Control y gestión de tareas: Permite gestionar y realizar un seguimiento de las tareas que realiza cada empleado. Pueden agregarse tareas, actualizarse, listar por departamento, listar por empleados o eliminarse. Las tareas, que se generan de forma automática cuando se da de alta el detalle de un pedido, lo hacen sin

ser asignadas a ningún empleado. El encargado podrá determinar qué operario se adjudicará a cada tarea. En el caso que esa persona tenga demasiado trabajo, podrá rechazarla y otro empleado del mismo departamento podrá aceptarla, hecho mediante el cual se “reasigna” la tarea. El software brinda la posibilidad de consultar las tareas que tiene asignadas cada empleado.

7. Gestión de clientes: Posibilita el mantenimiento de una base de datos de clientes, con opciones para añadir nuevos clientes, consultar detalles, modificar sus datos y eliminarlos según sea necesario.
8. Gestión de empleados: Permite mantener un registro completo de todos los empleados de la fábrica, con opciones para añadir nuevos empleados, consultar detalles, modificar datos y eliminar empleados según sea necesario.
 - 8.1. El administrador puede visualizar los pedidos gestionados por cada empleado en un rango entre fechas.
 - 8.2. El administrador puede consultar los empleados que pertenecen a un determinado departamento.
 - 8.3. El administrador puede visualizar qué empleados se encuentran activos y cuáles están desvinculados de la empresa.
9. Todas las vistas tienen un buscador que permite ingresar cualquier parámetro de la tabla para realizar la búsqueda.

Conclusiones y mejoras del proyecto

El desarrollo de este proyecto de software para la gestión de fábricas de sofás ha sido un proceso meticuloso y centrado en el usuario, culminando en la creación de una aplicación web diseñada para mejorar la eficiencia y la productividad en el sector. Desde la fase inicial de investigación y diseño hasta la implementación y pruebas finales, se ha puesto un énfasis considerable en comprender las necesidades de los usuarios y proporcionar soluciones que aborden sus desafíos específicos.

La aplicación desarrollada ofrece una amplia gama de funcionalidades, incluyendo la gestión de materiales y stock, el control de proveedores, un catálogo de modelos de sofás con sus respectivos materiales, el seguimiento detallado de pedidos y, la gestión de clientes y empleados. Cada una de estas características ha sido cuidadosamente diseñada y probada para garantizar una experiencia de usuario fluida y satisfactoria.

Además, se ha prestado especial atención al diseño del frontend, siguiendo principios de usabilidad y experiencia de usuario, así como a la implementación de un sistema de diseño que considera los modelos mentales de los usuarios. Esto ha resultado en una interfaz intuitiva y coherente, que facilita la interacción y mejora la eficacia de la aplicación.

El uso de metodologías como el Design Thinking y el Atomic Design ha permitido integrar la retroalimentación de los usuarios y optimizar el proceso de desarrollo, asegurando que la aplicación final satisfaga las necesidades y expectativas del mercado.

En lo que respecta a la implementación de mejoras, en primer lugar, se podría explorar la integración de tecnologías emergentes, como la inteligencia artificial y el aprendizaje automático, para optimizar aún más los procesos de producción, prever la demanda de productos y ofrecer recomendaciones personalizadas a los clientes.

Otro aspecto que resulta fundamental y que no se ha podido abordar por cuestiones de tiempo es la seguridad. En particular, el registro de sesiones por usuario resulta primordial en este tipo de aplicaciones. Este registro permite monitorear y controlar el acceso a la información, garantizando que solo los usuarios autorizados puedan acceder a datos sensibles. Implementar un sistema de registro de sesiones no solo protege la

integridad de la aplicación, sino que también refuerza la confianza de los usuarios en la seguridad de sus datos personales.

Se quiere destacar también, que es importante mantener un ciclo de retroalimentación constante con los usuarios finales para identificar áreas de mejora y nuevas funcionalidades que puedan agregar valor al sistema. Esto podría lograrse mediante la implementación de mecanismos de retroalimentación directa en la aplicación y la organización regular de sesiones de usuario para recopilar comentarios y sugerencias.

De igual modo, se podría considerar la expansión de la aplicación para abarcar otros aspectos relacionados con la gestión empresarial, como la gestión financiera y la planificación de recursos empresariales, para proporcionar a las empresas una solución integral y completa.

En resumen, este proyecto representa un paso significativo hacia la modernización y la optimización de la gestión en el sector de la fabricación de sofás, proporcionando a las empresas herramientas poderosas y accesibles para mejorar su eficiencia operativa y su competitividad en el mercado.

Por último, no puede obviarse el hecho de que este proyecto demostró la importancia de un enfoque multidisciplinario que combina investigación, diseño y desarrollo para crear soluciones efectivas en el ámbito empresarial. El resultado final es una aplicación web escalable y fácil de usar que tiene el potencial de mejorar la eficiencia y la gestión en empresas del sector de fabricación de sofás.

Bibliografía

- ❖ Cerdá, R. (2023). *User eXperience* (Apuntes de asignatura). Universidad Internacional de la Rioja.
- ❖ Cerdá, R. (2023). *Usabilidad* (Apuntes de asignatura). Universidad Internacional de la Rioja.
- ❖ Cerdá, R. (2023). *Accesibilidad* (Apuntes de asignatura). Universidad Internacional de la Rioja.
- ❖ Cerdá, R. (2023). *Design Thinking* (Apuntes de asignatura). Universidad Internacional de la Rioja.
- ❖ Cordero, A. (2024). *Estudio de mercado*. (Apuntes de asignatura). Universidad Internacional de la Rioja.
- ❖ Salgado, Raúl. (2022). *Diseño de bases de datos relacional*. (Apuntes de asignatura). Universidad Internacional de la Rioja.

Páginas web

- ❖ Atlassian. (n.d.). Jira. Recuperado de <https://www.atlassian.com/es/software/jira>
- ❖ Bootstrap. (n.d.). The most popular HTML, CSS, and JS library in the world. Recuperado de <https://getbootstrap.com/>
- ❖ Canva. (n.d.). Canva: Diseño gráfico para educadores y estudiantes. Recuperado de <https://www.canva.com/>
- ❖ De Pablo, F. (2023). *HyperText Markup Language (HTML)*. Recuperado de <https://github.com/fdepablo/WsLMSGI/blob/master/README.md>
- ❖ De Pablo, F. (2023). *Javadoc*. Recuperado de https://github.com/fdepablo/WorkspaceJava/tree/master/20_JavaDoc
- ❖ De Pablo, F. (2023). *Git*. Recuperado de <https://github.com/fdepablo/WsGit>
- ❖ Herrera, F. (2024). Vue.js de Cero a experto. Udemy. Recuperado el 02 de mayo de 2024, de <https://www.udemy.com/course/vuejs-de-cero-a-experto>
- ❖ draw.io. (s.f.). draw.io. Recuperado de <https://www.draw.io>
- ❖ Figma. (n.d.). Figma: The collaborative interface design tool. Recuperado de <https://www.figma.com/>
- ❖ Font Awesome. (n.d.). Font Awesome: The web's most popular icon set and toolkit. Recuperado de <https://fontawesome.com/>

- ❖ GitHub. (n.d.). GitHub. Recuperado de <https://github.com/>
- ❖ PhotoRoom. (n.d.). Edición de fotos con IA: elimina los fondos y crea imágenes de productos Recuperado de <https://www.photoroom.com/es>
- ❖ Spring. (n.d.). Spring Boot reference documentation. Recuperado el 02 de abril de 2024, de <https://docs.spring.io/spring-boot/documentation.html>
- ❖ Unsplash. (n.d.). Hermosas imágenes y fotos gratuitas. Recuperado de <https://unsplash.com/es>
- ❖ Vue.js. (s. f.). Guía de Vue.js. Recuperado el 29 de abril de 2024 de [Introducción — Vue.js \(vuejs.org\)](#)

Anexos

Código fuente de la aplicación

Método que devuelve el estado general de un pedido

```
@GetMapping("/estadoPorPedido/{idPedido}")
public ResponseEntity<?> mostrarEstadoPedido(@PathVariable int idPedido) {
    List<DetallePedido> lista = detallePedidoService.buscarPorIdPedido(idPedido);

    if (lista == null || lista.isEmpty()) {
        return ResponseEntity.status(HttpStatus.OK).body("");
    }

    boolean todasTareasFinalizadas = true;
    boolean todasTareasSinAsignar = true;
    boolean pendiente = false;
    boolean procesando = false;
    boolean cancelada = false;

    for (DetallePedido detalle : lista) {
        List<Tarea> tareas = tareaService.buscarPorDetalle(detalle.getIdDePed());

        if (tareas == null || tareas.isEmpty()) {
            return ResponseEntity.status(HttpStatus.NOT_FOUND).body("-");
        }
    }
```

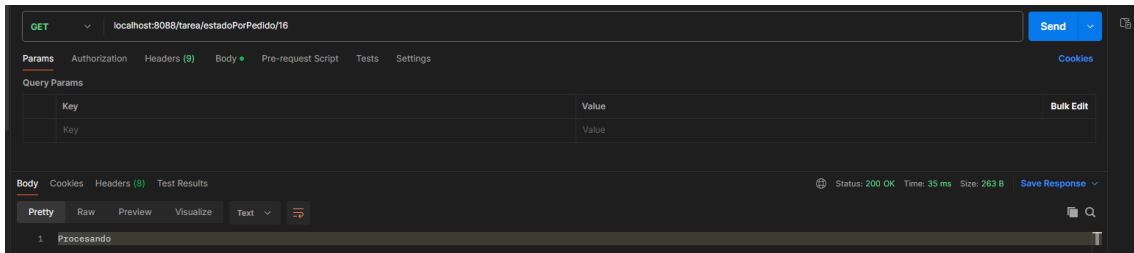
```
    for (DetallePedido detalle : lista) {
        List<Tarea> tareas = tareaService.buscarPorDetalle(detalle.getIdDePed());

        if (tareas == null || tareas.isEmpty()) {
            return ResponseEntity.status(HttpStatus.NOT_FOUND).body("-");
        }

        for (Tarea tarea : tareas) {
            int estado = tarea.getEstado().getIdEstado();

            if (estado == 1) {
                pendiente = true;
                todasTareasSinAsignar = false;
                todasTareasFinalizadas = false;
            } else if (estado == 2) {
                procesando = true;
                todasTareasSinAsignar = false;
                todasTareasFinalizadas = false;
            } else if (estado != 3) {
                todasTareasFinalizadas = false;
            } else if (estado == 4) {
                cancelada = true;
                todasTareasFinalizadas = false;
            } else if (estado != 5) {
                todasTareasSinAsignar = false;
            }
        }
    }
```

```
    if (todasTareasSinAsignar) {
        return ResponseEntity.ok("Sin asignar");
    } else if (cancelada) {
        return ResponseEntity.ok("Cancelado");
    } else if (procesando) {
        return ResponseEntity.ok("Procesando");
    } else if (pendiente) {
        return ResponseEntity.ok("Pendiente");
    } else if (todasTareasFinalizadas) {
        return ResponseEntity.ok("Finalizado");
    } else {
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body("Unexpected error");
    }
}
```



Método que asigna un empleado a una tarea

```
@PostMapping("/asignarEmpleado/{idTarea}")
public ResponseEntity<?> asignarEmpleado(@PathVariable(name = "idTarea") int idTarea,
                                           @RequestParam(name = "idEmpleado") int idEmpleado){

    try {

        Tarea tarea = tareaService.buscarTarea(idTarea);
        DetallePedido detallePedido = detallePedidoService.buscarDetPed(tarea.getDetalle().getIdDePed());
        //Comprobar si la tarea existe y tiene detalle
        if (detallePedido != null && tarea != null) {

            Empleado empleado = empleadoService.buscarUno(idEmpleado);

            //Comprobar si el empleado existe
            if (empleado != null) {
                tarea.setEmpleado(empleado);
                tarea.setEstado(estadosService.buscarEstado(1));
                tareaService.modifTarea(tarea);
            }
            else {
                return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR)
                    .body("El empleado no existe");
            }
        }
        return ResponseEntity.status(HttpStatus.OK).body(tarea);

    } catch (Exception e) {
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR)
            .body("Error al asignar un empleado a la tarea: " + e.getMessage());
    }
}
```

PUT localhost:8088/tarea/asignarEmpleado/8?idEmpleado=2

Status: 200 OK Time: 142 ms Size: 1.4 KB Save Response

```
{
  "idTarea": 8,
  "empleado": {
    "idEmpleado": 2,
    "nombre": "Marta",
    "apellidos": "Garcia",
    "password": "maria",
    "departamento": {
      "idDepartamento": 2,
      "nombre": "costura"
    }
  }
}
```

Método que da de alta a un material

```
50 @Test
51 public void testAlta() {
52     // Crea un material de ejemplo
53     MaterialDto nuevoMaterialDto = new MaterialDto();
54     nuevoMaterialDto.setNombre("Grapa"); // Establece el nombre del material
55     nuevoMaterialDto.setDescripcion("Grapas metálicas de acero"); // Establece una descripción para el material
56     nuevoMaterialDto.setCantidad(50); // Establece la cantidad de ese material
57     nuevoMaterialDto.setRefMaterialProveedor(3); // Establece la referencia del material del proveedor
58     nuevoMaterialDto.setCategoria("textil"); // Establece la categoría del material
59     nuevoMaterialDto.setUnidadMedida("unidad"); // Establece la unidad del material
60     nuevoMaterialDto.setIdProveedor(1); // Establece el ID del proveedor del material
61
62     // llama al método "altaMaterial"
63     ResponseEntity<?> responseEntity = materialRestController.altaMaterial(nuevoMaterialDto);
64
65     // Verifica que el código de estado de la respuesta sea OK (200)
66     assertEquals(HttpStatus.OK, responseEntity.getStatusCode());
67
68     // Obtiene el material guardado del cuerpo de la respuesta
69     Material materialGuardado = (Material) responseEntity.getBody();
70
71     // Verifica que el material guardado no sea nulo
72     assertNotNull(materialGuardado, "El material guardado no debería ser nulo");
73
74     // Verifica que el nombre del material guardado coincida con el nombre esperado
75     assertEquals("Grapa", materialGuardado.getNombre(), "Error al ingresar el material");
76 }
77 }
78
```

Test con JUnit

```

:: Spring Boot :: (v3.2.4)

2024-05-29T19:42:34.688+02:00 INFO 5052 --- [main] r.r.m.MaterialRestControllerTestAlta : Starting MaterialRestControllerTestAlta using
2024-05-29T19:42:34.690+02:00 INFO 5052 --- [main] r.r.m.MaterialRestControllerTestAlta : No active profile set, falling back to 1 default
2024-05-29T19:42:35.729+02:00 INFO 5052 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in
2024-05-29T19:42:35.818+02:00 INFO 5052 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 7
2024-05-29T19:42:36.289+02:00 INFO 5052 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2024-05-29T19:42:36.643+02:00 INFO 5052 --- [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.
2024-05-29T19:42:36.644+02:00 INFO 5052 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2024-05-29T19:42:36.685+02:00 INFO 5052 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH0000204: Processing PersistenceUnitInfo [na
2024-05-29T19:42:36.721+02:00 INFO 5052 --- [main] org.hibernate.Version : HHH0000412: Hibernate ORM core version 6.4.4.F
2024-05-29T19:42:36.785+02:00 INFO 5052 --- [main] o.h.c.internal.RegionFactoryInitiator : HHH000026: Second-level cache disabled
2024-05-29T19:42:37.146+02:00 INFO 5052 --- [main] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup: ignoring JPA class t
2024-05-29T19:42:37.226+02:00 WARN 5052 --- [main] org.hibernate.orm.deprecation : HHH0000025: MySQLDialect does not need to be
2024-05-29T19:42:38.040+02:00 INFO 5052 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hi
2024-05-29T19:42:38.043+02:00 INFO 5052 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for pers
2024-05-29T19:42:38.369+02:00 INFO 5052 --- [main] o.s.d.j.r.query.QueryEnhancerFactory : Hibernate is in classpath; If applicable, HQL
2024-05-29T19:42:38.966+02:00 WARN 5052 --- [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default
2024-05-29T19:42:39.400+02:00 INFO 5052 --- [main] r.r.m.MaterialRestControllerTestAlta : Started MaterialRestControllerTestAlta in 4.9
Hibernate: select p1_0.id proveedor,p1_0.descripcion,p1_0.nombre,p1_0.telefono from proveedores p1_0 where p1_0.id proveedor=?
Hibernate: insert into materiales (cantidad,categoria,descripcion,nombre,id proveedor,ref_material_prov,unidad medida) values (?, ?, ?, ?, ?, ?)
2024-05-29T19:42:41.512+02:00 INFO 5052 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persiste
2024-05-29T19:42:41.516+02:00 INFO 5052 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...
2024-05-29T19:42:41.531+02:00 INFO 5052 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.
```

Scripts de ejecución/instalación o de cargas de BBDD

```
1
2 • CREATE DATABASE IF NOT EXISTS sofa_bbdd;
3 • use sofa_bbdd;
4
5 -- tables
6 -- Table: clientes
7 • CREATE TABLE clientes (
8     id_cliente int NOT NULL AUTO_INCREMENT,
9     nombre varchar(100) NULL DEFAULT null ,
10    apellidos varchar(150) NULL DEFAULT null,
11    direccion varchar(300) NULL DEFAULT null,
12    email varchar(100) NULL DEFAULT null,
13    telefono varchar(20) NULL DEFAULT null,
14    CONSTRAINT clientes_pk PRIMARY KEY (id_cliente)
15 );
16
17 -- Table: departamentos
18 • CREATE TABLE departamentos (
19     id_depto int NOT NULL AUTO_INCREMENT,
20     nombre varchar(50) NULL DEFAULT null,
21     CONSTRAINT departamentos_pk PRIMARY KEY (id_depto)
22 );
23
24 -- Table: detalle_pedido
25 • CREATE TABLE detalle_pedido (
26     id_deped int NOT NULL AUTO_INCREMENT,
27     id_pedido int NULL DEFAULT null,
28     id_sofa int NULL DEFAULT null,
29     cantidad int NULL DEFAULT null,
30     plazas int NULL DEFAULT null,
31     dens_cojin decimal(10,2) NULL DEFAULT null,
32     fecha date NULL DEFAULT null,
33     precio decimal(10,2) NULL DEFAULT null,
34     CONSTRAINT detalle_pedido_pk PRIMARY KEY (id_deped)
35 );
36
37 -- Table: empleado_departamento
38 • CREATE TABLE empleado_departamento (
39     id_ed int NOT NULL AUTO_INCREMENT,
40     id_empleado int NULL DEFAULT null,
41     id_depto int NULL DEFAULT null,
42     CONSTRAINT empleado_departamento_pk PRIMARY KEY (id_ed)
43 );
```

```

45  -- Table: empleados
46  ● CREATE TABLE empleados (
47      id_empleado int NOT NULL AUTO_INCREMENT,
48      nombre varchar(100) NULL DEFAULT null,
49      apellidos varchar(100) NULL DEFAULT null,
50      password varchar(100) NULL DEFAULT null,
51      id_depto int NULL DEFAULT null,
52      id_perfil int NULL DEFAULT null,
53      fecha_ingreso date NULL DEFAULT null,
54      fecha_baja date NULL DEFAULT null,
55      estado varchar(15) NULL DEFAULT null,
56      salario decimal(10,2) NULL DEFAULT null,
57      CONSTRAINT empleados_pk PRIMARY KEY (id_empleado)
58  );
59
60  -- Table: estados
61  ● CREATE TABLE estados (
62      id_estado int NOT NULL AUTO_INCREMENT,
63      nombre varchar(50) NULL DEFAULT null,
64      CONSTRAINT estados_pk PRIMARY KEY (id_estado)
65  );
66
67  -- Table: material_proveedor
68  ● CREATE TABLE material_proveedor (
69      id_mat_prov int NOT NULL AUTO_INCREMENT,
70      id_proveedor int NULL DEFAULT null,
71      id_material int NULL DEFAULT null,
72      CONSTRAINT material_proveedor_pk PRIMARY KEY (id_mat_prov)
73  );
74
75  -- Table: materiales
76  ● CREATE TABLE materiales (
77      id_material int NOT NULL AUTO_INCREMENT,
78      nombre varchar(100) NULL DEFAULT null,
79      descripcion varchar(100) NULL DEFAULT null,
80      cantidad decimal(10,2) NULL DEFAULT null,
81      id_proveedor int NULL DEFAULT null,
82      ref_material_prov int NULL DEFAULT null,
83      categoria varchar(100) NULL DEFAULT null,
84      unidad_medida varchar(100) NULL DEFAULT null,
85      CONSTRAINT materiales_pk PRIMARY KEY (id_material)
86  );
87
88  -- Table: pedidos
89  ● CREATE TABLE pedidos (
90      id_pedido int NOT NULL AUTO_INCREMENT,
91      id_cliente int NULL DEFAULT null,
92      fecha date NULL DEFAULT null,
93      vendedor int NULL DEFAULT null,
94      CONSTRAINT pedidos_pk PRIMARY KEY (id_pedido)
95  );
96

```

```

98      -- Table: perfiles
99  ● ○ CREATE TABLE perfiles (
100      id_perfil int NOT NULL AUTO_INCREMENT,
101      rol varchar(50) NULL DEFAULT null,
102      CONSTRAINT perfiles_pk PRIMARY KEY (id_perfil)
103  );
104
105      -- Table: proveedores
106  ● ○ CREATE TABLE proveedores (
107      id_proveedor int NOT NULL AUTO_INCREMENT,
108      nombre varchar(100) NULL DEFAULT null,
109      telefono int NULL DEFAULT null,
110      descripcion varchar(300) NULL DEFAULT null,
111      CONSTRAINT proveedores_pk PRIMARY KEY (id_proveedor)
112  );
113
114      -- Table: sofa_materiales
115  ● ○ CREATE TABLE sofa_materiales (
116      id_sm int NOT NULL AUTO_INCREMENT,
117      id_sofa int NULL DEFAULT null,
118      id_material int NULL DEFAULT null,
119      cantidad_utilizada int NULL DEFAULT null,
120      CONSTRAINT sofa_materiales_pk PRIMARY KEY (id_sm)
121  ) ;
122
123
124
125
126      -- Table: sofas
127
128  ● ○ CREATE TABLE sofas (
129      id_sofa int NOT NULL AUTO_INCREMENT,
130      nombre varchar(100) NULL DEFAULT null,
131      descripcion varchar(300) NULL DEFAULT null,
132      patas int NULL DEFAULT null,
133      medida_cojin varchar(100) NULL DEFAULT null,
134      precio decimal(10,2) NULL DEFAULT null,
135      CONSTRAINT sofas_pk PRIMARY KEY (id_sofa)
136  );
137
138      -- Table: tareas
139  ● ○ CREATE TABLE tareas (
140      id_tarea int NOT NULL AUTO_INCREMENT,
141      id_empleado int NULL DEFAULT null,
142      id_depto int NULL DEFAULT null,
143      id_estado int NULL DEFAULT null,
144      fecha date NULL DEFAULT null,
145      id_deped int NOT NULL,
146      CONSTRAINT tareas_pk PRIMARY KEY (id_tarea)
147  );

```



```

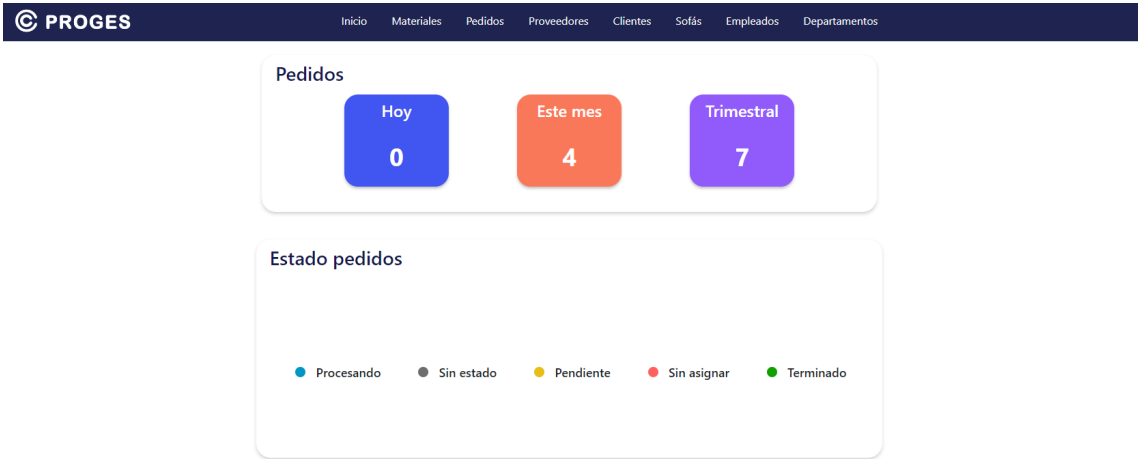
157 -- foreign keys
158 -- Reference: empleados_ibfk_2 (table: empleados)
159 • ALTER TABLE empleados ADD CONSTRAINT empleados_ibfk_2 FOREIGN KEY empleados_ibfk_2 (id_perfil)
160     REFERENCES perfiles (id_perfil);
161
162 -- Reference: id_cliente (table: pedidos)
163 • ALTER TABLE pedidos ADD CONSTRAINT id_cliente FOREIGN KEY id_cliente (id_cliente)
164     REFERENCES clientes (id_cliente);
165
166 -- Reference: id_depto3 (table: tareas)
167 • ALTER TABLE tareas ADD CONSTRAINT id_depto3 FOREIGN KEY id_depto3 (id_depto)
168     REFERENCES departamentos (id_depto);
169
170 -- Reference: id_depto6 (table: empleado_departamento)
171 • ALTER TABLE empleado_departamento ADD CONSTRAINT id_depto6 FOREIGN KEY id_depto6 (id_depto)
172     REFERENCES departamentos (id_depto);
173
174 -- Reference: id_empleado (table: pedidos)
175 • ALTER TABLE pedidos ADD CONSTRAINT id_empleado FOREIGN KEY id_empleado (vendedor)
176     REFERENCES empleados (id_empleado);
177
178 -- Reference: id_empleado2 (table: tareas)
179 • ALTER TABLE tareas ADD CONSTRAINT id_empleado2 FOREIGN KEY id_empleado2 (id_empleado)
180     REFERENCES empleados (id_empleado);
181
182 -- Reference: id_empleado6 (table: empleado_departamento)
183 • ALTER TABLE empleado_departamento ADD CONSTRAINT id_empleado6 FOREIGN KEY id_empleado6 (id_empleado)
184     REFERENCES empleados (id_empleado);
185
186 -- Reference: id_estado2 (table: tareas)
187 • ALTER TABLE tareas ADD CONSTRAINT id_estado2 FOREIGN KEY id_estado2 (id_estado)
188     REFERENCES estados (id_estado);
189
190 -- Reference: id_material1 (table: sofa_materiales)
191 • ALTER TABLE sofa_materiales ADD CONSTRAINT id_material1 FOREIGN KEY id_material1 (id_material)
192     REFERENCES materiales (id_material);
193
194 -- Reference: id_material3 (table: material_proveedor)
195 • ALTER TABLE material_proveedor ADD CONSTRAINT id_material3 FOREIGN KEY id_material3 (id_material)
196     REFERENCES materiales (id_material);
197
198 -- Reference: id_pedido9 (table: detalle_pedido)
199 • ALTER TABLE detalle_pedido ADD CONSTRAINT id_pedido9 FOREIGN KEY id_pedido9 (id_pedido)
200     REFERENCES pedidos (id_pedido);
201
202 -- Reference: id_proveedor1 (table: material_proveedor)
203 • ALTER TABLE material_proveedor ADD CONSTRAINT id_proveedor1 FOREIGN KEY id_proveedor1 (id_proveedor)
204     REFERENCES proveedores (id_proveedor);
205
206 -- Reference: id_sofa1 (table: sofa_materiales)
207 • ALTER TABLE sofa_materiales ADD CONSTRAINT id_sofa1 FOREIGN KEY id_sofa1 (id_sofa)
208     REFERENCES sofas (id_sofa);
209
210
211 • ALTER TABLE detalle_pedido ADD CONSTRAINT id_sofa9 FOREIGN KEY id_sofa9 (id_sofa)
212     REFERENCES sofas (id_sofa);
213
214 -- Reference: tareas_detalle_pedido (table: tareas)
215 • ALTER TABLE tareas ADD CONSTRAINT tareas_detalle_pedido FOREIGN KEY tareas_detalle_pedido (id_deped)
216     REFERENCES detalle_pedido (id_deped);

```

Vistas de la aplicación

Vista de inicio

Vistas



Vistas de elementos del menú

Lista de sofás									
<div>Buscar...</div>									
Nombre	Descripción	Patas	Medida de cojín	Precio					
Aithara	Sofá de estilo clásico con tapicería de cuero	4	50.00	500	Editar	Material	Eliminar		
Lucia	Chaise longue izquierdo	4	45.00	700	Editar	Material	Eliminar		
Luna	Chaise longue derecho	4	48.00	600	Editar	Material	Eliminar		
Sevilla	Simple 2 y 3 plazas	6	55.00	800	Editar	Material	Eliminar		
Iris	Respaldo reclinable en 2 y 3 plazas	4	60.00	900	Editar	Material	Eliminar		
Benedetta	sofá Chesterfield	4	30	1200	Editar	Material	Eliminar		
Positano	Sofá de estilo chaise long tapizado en gamupiel	6	60.00	2000	Editar	Material	Eliminar		
Perlea	Sofá de estilo moderno tapizado en cuero sintético	4	50.00	1400	Editar	Material	Eliminar		

© PROGES

Inicio

Materiales

Pedidos

Proveedores

Clientes

Sofás

Empleados

Departamentos

Lista de sofás

Nuevo

Buscar...

Nombre	Descripción	Patas	Medida de cojín	Precio			
Aithara	Sofá de estilo clásico con tapicería de cuero	4	50.00	500	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>
Lucía	Chaise longue izquierdo	4	45.00	700	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>
Luna	Chaise longue derecho	4	48.00	600	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>
Sevilla	Simple 2 y 3 plazas	6	55.00	800	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>
Iris	Respaldo reclinable en 2 y 3 plazas	4	60.00	900	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>
Benedetta	sofá Chesterfield	4	30	1200	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>
Positano	Sofá de estilo chase long tapizado en gamupiel	6	60.00	2000	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>
Perlea	Sofá de estilo moderno tapizado en cuero sintético	4	50.00	1400	<div>Editar</div>	<div>Material</div>	<div>Eliminar</div>

Vista de los materiales necesarios para producir un determinado modelo de sofá

© PROGES

Inicio

Materiales

Pedidos

Proveedores

Clientes

Sofás

Empleados

Departamentos

Materiales ref-sofa 1

Nuevo

Referencia material	Referencia proveedor	Nombre proveedor	Nombre material	Descripción	Cantidad	Unidad medida		
1	456	Suministros Vanguardia	Tornillos	Tornillos madera 7cm	20	caja	<div>Editar</div>	<div>Eliminar</div>
2	357	Excelencia	Guata	Guata blanca 50cm ancho	5		<div>Editar</div>	<div>Eliminar</div>
3	100	ACME	Hilo	Hilo costura	2	bobina	<div>Editar</div>	<div>Eliminar</div>
4	757	ACME	Tela	Rivera beige	14	m2	<div>Editar</div>	<div>Eliminar</div>
6	112	Maderas Moral	Madera	Madera pino esqueleto	5	ml	<div>Editar</div>	<div>Eliminar</div>
7	654	Excelencia	Cojines	Cojines sofas	5	unidad	<div>Editar</div>	<div>Eliminar</div>
8	1795	Avanza Global	Cinchas	Cinchas duras	5	ml	<div>Editar</div>	<div>Eliminar</div>
11	489	Innova	Patas	Patas acero inox rectas	4	unidad	<div>Editar</div>	<div>Eliminar</div>

Vista de los departamentos

© PROGES

Inicio

Materiales

Pedidos

Proveedores

Clientes

Sofás

Empleados

Departamentos

Departamentos

Carpintería

Costura

Tapizado

Enfundado

9

4

1

1

Sin asignar

Pendiente

Procesando

Terminado

Buscar...

Fecha creación	Referencia tarea	Número pedido	Ref detalle	Empleado					
26-05-2024	7	16	7	Luis	<div>Asignar</div>	<div>Desvincular</div>	<div>Procesando</div>	<div>Terminado</div>	<div>Cancelar</div>
26-05-2024	11	16	8	Laura	<div>Asignar</div>	<div>Desvincular</div>	<div>Procesando</div>	<div>Terminado</div>	<div>Cancelar</div>
26-05-2024	15	16	9	Paco	<div>Asignar</div>	<div>Desvincular</div>	<div>Procesando</div>	<div>Terminado</div>	<div>Cancelar</div>
28-05-2024	23	17	11	María	<div>Asignar</div>	<div>Desvincular</div>	<div>Procesando</div>	<div>Terminado</div>	<div>Cancelar</div>

Formularios de alta

PROGES Inicio Materiales Pedidos Proveedores Clientes Sofás Empleados Departamentos

Formulario de alta de modelos de sofá

Nombre	Descripción
<input type="text"/>	<input type="text"/>
Patas	Medida cojín
<input type="text"/>	<input type="text"/>
Precio	
<input type="text"/>	

Formularios de edición

PROGES Inicio Materiales Pedidos Proveedores Clientes Sofás Empleados Departamentos

Actualizar datos del proveedor

Nombre material	Teléfono
<input type="text" value="ACME"/>	<input type="text" value="682659477"/>
Descripción	
<input type="text" value="Cantidad"/>	

Procedimiento para desplegar la aplicación

Requisitos Previos Java Development Kit (JDK) 11 o superior

- ❖ Descargar JDK Eclipse IDE for Java Developers
- ❖ Descargar Eclipse Maven
- ❖ Instalar Maven Node.js y npm
- ❖ Descargar Node.js Visual Studio Code
- ❖ Descargar Visual Studio Code MySQL Server y MySQL Workbench
- ❖ Descargar MySQL

Instrucciones de Configuración

1. Clonar el Repositorio

<https://github.com/anabella-aceto/ProyectoFinal.git>

2. Configuración del Backend Importar el Proyecto en Eclipse Abrir Eclipse. Seleccionar File -> Import. Elegir Existing Maven Projects y hacer clic en Next. Buscar la ubicación del proyecto clonado y seleccionar la carpeta raíz del proyecto. Eclipse debería detectar automáticamente el archivo pom.xml. Hacer clic en Finish.

Configurar la Base de Datos

1. Abrir MySQL Workbench y conectarse a su servidor MySQL.
2. Descargar y ejecutar el script SQL proporcionado (script.sql) en la nueva base de datos. Para ello, abrir el script en MySQL Workbench y ejecutarlo.
3. Configurar el Archivo application.properties

En el directorio src/main/resources, abrir el archivo application.properties y configurar las propiedades de conexión a la base de datos:

4. Compilar y Ejecutar el Backend

En Eclipse, abrir el archivo principal de la aplicación (normalmente se encuentra en src/main/java/com/tu/paquete/Application.java). Hacer clic derecho en el archivo y seleccionar Run As → Spring Boot App.

Configuración del Frontend

1. Clonar el repositorio

<https://github.com/Daviid7505/sofa-gestion.git>

2. Importar el Proyecto en Visual Studio Code

- a. Abrir Visual Studio Code.
- b. Seleccionar File → Open Folder.
- c. Navegar a la carpeta del proyecto clonado y abrir la carpeta frontend.

3. Instalar Dependencias

- a. Abrir una terminal en Visual Studio Code y ejecutar: “npm install”
- b. Luego, ejecutar el comando: “npm run serve”

4. Verificar el Despliegue

Abrir un navegador web y navegar al puerto que se indica en la consola de comandos de VSC para verificar que el frontend esté funcionando.