

T2-SGE-SISTEMAS-DE-GESTION-EMPRES...



bloodyraintatii



Sistemas de Gestión Empresarial



1º Desarrollo de Aplicaciones Multiplataforma



Estudios España



Accede al documento original

antes



**Descarga sin publi
con 1 coin**



Después

WUOLAH



Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

perdo
espacio



Necesito
concentración

ali ali ooh
esto con 1 coin me
lo quito yo...

WUOLAH

Sistemas de Gestión Empresarial

Tema 10: Adecuación de un sistema ERP-CRM a una empresa

ACTUALIZACIÓN DE ODOO (no?)

- Se recomienda actualizar OdoO periódicamente para obtener mejoras de rendimiento, correcciones de errores y actualizaciones de seguridad.
- Se debería hacer un backup de la bbdd antes de actualizar.
- Se puede actualizar mediante:
 - Un paquete instalador → descarga e instala la nueva versión sobre la actual.
 - Con archivos de código fuente (GitHub) → se actualiza por comandos. Se pueden generar conflictos si se ha modificado el código.
 - Docker → se ejecuta una nueva versión copiando los archivos necesarios.

ADAPTANDO ODOO A LA EMPRESA (sí)

- Personalización básica:
 - Modificar plantillas de documentos (facturas, presupuestos...)
 - Cambiar plantillas de correos electrónicos.
 - Agregar el logo de la empresa y actualizar la información general.
- Herramientas:
 - Ajustes/opciones generales → para modificar información de la compañía y plantillas de documentos.
 - Menú técnico/correo electrónico → para editar plantillas de correos.

MODIFICACIÓN DE LOS MÓDULOS DEL SISTEMA CON ODOO (puede pero 4b/la parte de informes no? 3d, 3e, 3h)

- OdoO Studio: Herramienta de OdoO que permite personalizar formularios, informes, vistas (kanban, gráficos, etc.) sin necesidad de programar.
- Funcionalidades:
 - Crear y modificar aplicaciones.
 - Diseñar interfaces amigables para móviles.
 - Usar una interfaz gráfica de arrastrar y soltar.
- Limitación → solo disponible en la versión Enterprise (no es Open Source).

PERSONALIZANDO UN FORMULARIO EXISTENTE CON ODOO STUDIO (CASO PRÁCTICO) (no? 4b, 4d)

- Objetivo → modificar la vista 'formulario' del módulo de empleados en OdoO para añadir una nueva pestaña que permita gestionar si el empleado tiene hijos y, en caso afirmativo, cuántos.
- Pasos:
 1. Abrir OdoO Studio desde el módulo de empleados.
 2. Seleccionar "Vistas" y luego "Formularios".
 3. Agregar una nueva pestaña llamada "Hijos".

WUOLAH

4. Añadir un campo de tipo 'casilla de verificación' para la pregunta '¿Tiene hijos?'.
5. Añadir un campo desplegable para '¿Cuántos?' con opciones de 1 a 5 y 'más'.
6. Configurar el desplegable como obligatorio cuando la casilla esté activada.

CREACIÓN DE MÓDULOS PROPIOS: CREACIÓN Y DISEÑO (no sé? 4b, 4d)

→ Creación de un módulo llamado "Gestión de alquileres" para una empresa de alquiler de oficinas.

→ Pasos:

1. Crear el módulo desde Odoo Studio, asignándole un nombre y un icono.
2. Diseñar el formulario con campos, como: nombre de la oficina, foto, precio del alquiler, número de puestos de trabajo, dirección.

CREACIÓN DE MÓDULOS PROPIOS: GESTIÓN Y VISTAS (no sé? 4b, 4d)

→ Barra de estado → agregar estados como "disponible", "alquilado" y "a renovar" para gestionar el alquiler de oficinas.

→ Vista de lista → mejorar la vista de lista agregando más campos (precio, dirección, etc.)

→ Vista kanban → crear una vista kanban para visualizar rápidamente el estado de las oficinas (alquiladas o disponibles).

AGREGANDO UNA VISTA KANBAN (CASO PRÁCTICO) (no? 4b)

→ Objetivo → crear una vista Kanban en el nuevo módulo de Gestión de Alquileres para visualizar rápidamente las oficinas que están alquiladas y las que no.

→ Pasos:

1. Acceder al módulo Gestión de Alquileres y abrir Odoo Studio.
2. Seleccionar "Vistas" y luego "Kanban".
3. Agrupar las propiedades por su estado (disponible, alquilado, a renovar) usando la barra de estado.
4. Guardar y visualizar la vista Kanban, que permitirá distinguir rápidamente las oficinas alquiladas de las disponibles.

CREAR INFORMES ESTADÍSTICOS (no? 3d, 3e, 3h)

→ Objetivo → crear informes estadísticos y gráficos en Odoo Studio para analizar datos de manera visual, como el número de puestos de trabajo por oficina.

→ Pasos:

1. Acceder al módulo Gestión de Alquileres y abrir Odoo Studio.
2. Seleccionar "Vistas" y luego "Gráficos".
3. Configurar el gráfico para mostrar los datos deseados (ej: número de puestos de trabajo por oficina).
4. Aplicar filtros y opciones de visualización según sea necesario.
5. Crear un informe en blanco arrastrando campos como foto, nombre, dirección, precio de alquiler y número de puestos.
6. Guardar y visualizar el informe generado.

Tema 11: Lenguaje de programación de un sistema ERP-CRM: Python

¿QUÉ ES PYTHON?

→ Definición: Python es un lenguaje de programación creado por Guido van Rossum a finales de los 80. Su nombre proviene de los Monty Python, un grupo de humoristas británicos.

→ Características:

- Interpretado: No necesita compilación.
- Multiparadigma: Soporta programación orientada a objetos, funcional y procedural.
- De alto nivel: Fácil de leer y escribir.
- Tipado dinámico y fuerte: No es necesario declarar el tipo de variable, pero no permite operaciones entre tipos incompatibles.

→ Ventajas:

- Open source y multiplataforma (Windows, Mac, Linux).
- Es parecido a leer y escribir en inglés.
- Gran cantidad de librerías estándar y de terceros (ej: TensorFlow, Pandas, Django).
- Es uno de los lenguajes más populares, superando a Java en 2019 según GitHub.

INSTALACIÓN DE PYTHON

→ Se puede instalar Python en diferentes sistemas operativos:

- Windows → descargar e instalar desde python.org.
- Mac → ya viene preinstalado, pero se puede instalar otra versión siguiendo los mismos pasos que en Windows.
- Linux → usar el administrador de paquetes para instalar otra versión.

→ Se pueden tener varias versiones de Python instaladas en el mismo sistema.

IDE PYCHARM

→ PyCharm → IDE (Entorno de Desarrollo Integrado) para Python.

→ Pasos para descargarlo e instalarlo:

1. Descargar e instalar PyCharm Community Edition desde jetbrains.com.
2. Crear un nuevo proyecto en PyCharm, seleccionando un entorno Virtualenv (aislado para el proyecto) o un intérprete existente.
3. Configurar el intérprete de Python en PyCharm para ejecutar scripts.

“HOLA MUNDO” (CASO PRÁCTICO)

→ Objetivo → crear un programa básico que imprima "Hola mundo" en la consola.

→ Pasos:

1. Crear un archivo .py en PyCharm.
2. Escribir el código `print("Hola mundo")`
3. Configurar el intérprete en PyCharm.
4. Ejecutar el programa y ver el resultado en la consola.

EL COMANDO PIP (???)

→ Pip Instalador de Paquetes o Pip Instalador de Python → sistema de gestión de paquetes para instalar paquetes escritos en Python.

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo espacio



Necesito concentración

ali ali oohh
esto con 1 coin me
lo quito yo...

WUOLAH

→ Para instalarlo → usar el comando `pip install <nombre_de_la_librería>` para instalar paquetes (ej: `pip install redis`).

PALABRAS RESERVADAS Y COMENTARIOS

→ 33 palabras reservadas → and, as, class, def (función), del (borrar un objeto), for, from (importar partes de un módulo), global (declarar variable global), if...

→ Comentarios → #de una línea o """de varias líneas""".

TIPOS DE DATOS, CASTING, PRINT E INPUT

→ Tipos de datos:

- Numéricos → int, float, complex (3 + 1j).
- Cadenas → str = ("hola mundo")
- Booleanos → bool (True o False)
- Listas (contenido mutable) → lista = ["dato1", "dato2"...]
- Tuplas (contenido inmutable) → tupla = ("dato1", "dato2"...)
- Rangos → range → rango = range(4)
- Diccionarios/json (almacenan clave-valor) → dic = {"nombre": "Tati", "edad": "27"}
- Sets → set (colección no ordenada ni indexada) → set = ("dato1", "dato2"...)

→ Casting → convertir un tipo de dato a otro usando funciones como int(), float(), str().

→ Funciones:

- print() → muestra texto en la consola.
- input() → captura la entrada del usuario (siempre devuelve un str).

VARIABLES, CONSTANTES Y ERRORES

→ Variables → contenedores para almacenar valores. No necesitan declaración explícita. Debe empezar por '_' o letras. Solo pueden tener '_', letras o números. Son sensibles a mayúsculas/minúsculas.

→ Constantes → no existen en Python pero se suelen declarar en mayúsculas.

→ Errores → de sintaxis (código mal escrito: falta un paréntesis) o de ejecución (errores durante la ejecución: sumar un número y un carácter).

"EL ERROR COMÚN" (CASO PRÁCTICO)

→ Objetivo: Crear un programa que sume y multiplique dos números ingresados por el usuario.

→ Pasos:

1. Capturar los valores con input().
2. Convertir los valores a int usando int() (casting).
3. Realizar la suma y multiplicación.
4. Mostrar los resultados con print().

OPERADORES

→ Operadores aritméticos: +, -, *, /, %, ** (exponente), // (división entera).

→ Operadores de asignación: =, +=, -=, *=, /=.

→ Operadores de comparación: ==, !=, >, <, >=, <=.

→ Operadores lógicos: and, or, not.

→ Operadores de identidad: is, is not (comparan objetos en memoria).

→ Operadores de pertenencia: in, not in (verifican si un elemento está en una colección).

→ Operadores bit a bit: &, |, ^, ~, <<, >> (trabajan con números binarios).

WUOLAH

Tema 12: Programando en Python

INDENTACIÓN

- Indentación → el sangrado o desplazamiento hacia la derecha de una o varias líneas de código.
- En Python, es obligatoria para definir bloques de código (if, for, while...)
- Reemplaza el uso de llaves {} o puntos y comas ; que se usan en otros lenguajes.

```
if condicion:  
    print("La condición es verdadera")  
# Bloque sin indentar por lo que está fuera del bloque condicional
```

FLUJO SECUENCIAL IF

- Flujo secuencial (IF) → ejecuta un bloque de código si se cumple una condición.

```
if condicion1:  
    # Bloque si condicion1 es verdadera  
elif condicion2:  
    # Bloque si condicion2 es verdadera  
else:  
    # Bloque si ninguna condición es verdadera
```

FLUJO DE PROGRAMACIÓN ITERATIVO / BUCLES ITERATIVOS

- Bucle while → *while condicion:*
- Bucle for → *for elemento in secuencia:*
- Sentencias de control → *break* (detiene el bucle) / *continue* (sata a la siguiente iteración).

RANGE (¿)

- Crea una secuencia de números

```
range(5) → [0, 1, 2, 3, 4]  
range(5, 10, 2) → [5, 7, 9]
```

CONTROL DE ERRORES

- Para manejar errores durante la ejecución → try/except/else/finally
- Con el comando *raise* se pueden indicar errores específicos.




























```
try:  
    # Bloque de código que puede generar un error  
except TipoDeError as e:  
    # Bloque que maneja el error  
else:  
    # Bloque que se ejecuta si no hay errores  
finally:  
    # Bloque que siempre se ejecuta
```

“EL NÚMERO ENTERO MAYOR Y MENOR DE UNA LISTA” (CASO PRÁCTICO)

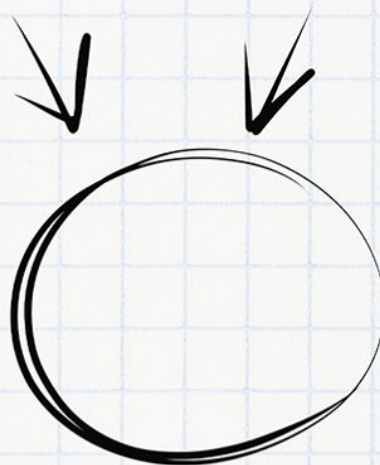
- Hay que verificar que los elementos sean un número antes de compararlos con la variable mayor y menor guardada.

Imagínate aprobando el examen

Necesitas tiempo y concentración

Planes	 PLAN TURBO	 PLAN PRO	 PLAN PRO+
 Descargas sin publi al mes	10 	40 	80 
 Elimina el video entre descargas			
 Descarga carpetas			
 Descarga archivos grandes			
 Visualiza apuntes online sin publi			
 Elimina toda la publi web			
 Precios Anual <input type="checkbox"/>	0,99 € / mes	3,99 € / mes	7,99 € / mes

Ahora que puedes conseguirlo,
¿Qué nota vas a sacar?



WUOLAH

ESTRUCTURA DEL CÓDIGO: FUNCIONES

→ Funciones → agrupan código reutilizable en bloques.

```
def sumar(a, b):  
    return a + b  
resultado = sumar(3, 4) # resultado = 7
```

ESTRUCTURA DEL CÓDIGO: MÓDULOS Y LIBRERÍAS

→ Módulos → archivos .py que contienen funciones y variables.

```
import modulo  
modulo.funcion()
```

→ Librerías → colecciones de módulos organizados en paquetes.

`pip install nombre_libreria` → instalación en bash

```
import libreria  
libreria.funcion()
```

→ uso en Python

FICHEROS

```
f = open("archivo.txt", "r") # Modo lectura → apertura  
r = lectura / a = agregar o crear si no existe / w = escritura / x = creación
```

```
contenido = f.read() → lectura
```

```
f.write("Texto nuevo") → escritura
```

```
f.close() → cierre
```

CADENAS DE TEXTO

→ Operadores comunes:

- Concatenación → `"Hola" + " mundo"`
- Subcadenas → `cadena[0:5]`
- Funciones → `cadena.upper()`, `cadena.split()`

CONVERSIÓN DE DECIMAL A BINARIO (CASO PRÁCTICO)

→ Se hace con un método propio que lo convierte y se guarda en un archivo.

ORIENTADO A OBJETOS

→ Clase → plantilla para crear objetos, que tienen propiedades (atributos) y comportamientos (métodos).

```
class Usuario:  
    def __init__(self, nombre, edad):  
        self.nombre = nombre  
        self.edad = edad  
    def mostrar_datos(self):  
        print(f"Nombre: {self.nombre}, Edad: {self.edad}")  
  
usuario = Usuario("Juan", 30)  
usuario.mostrar_datos()
```


Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo espacio



Necesito concentración

ali ali ooh
esto con 1 coin me
lo quito yo...

WUOLAH

Tema 14: Desarrollo de nuevos componentes I: creación de un módulo

CREACIÓN DE UN MÓDULO DESDE CERO

1. Crear un repositorio en una carpeta.
2. Agregar la ruta al `addon_path`.
3. Usar el comando `scaffold` para generar la estructura básica del módulo.
4. Configurar en el `__manifest__.py` el nombre, la descripción, las dependencias...

CONFIGURACIÓN Y VERIFICACIÓN DEL MÓDULO

1. Configurar `__manifest__.py` con `name`, `summary`, `description`, `depends`, `data`...
2. Actualizar la lista de aplicaciones (modo desarrollador necesario).
3. Instalar el módulo.
4. Verificar que funcione correctamente.

CREACIÓN DE UN MÓDULO CON HERENCIAS

1. Usar `scaffold` para generar la estructura del módulo.
2. Crear una clase que herede del modelo original usando `_inherit` en el archivo `models`.
3. Añadir campos adicionales al modelo heredado.
4. Heredar la vista → modificar la vista existente para incluir los nuevos campos.

"HERENCIA DEL MODELO" (CASO PRÁCTICO)

→ Objetivo: Agregar un campo "modo de pago" en el módulo de Ventas usando herencia.

→ Pasos:

1. Crear el módulo: Usar `scaffold` para generar el esqueleto del módulo.
2. Heredar el modelo `sale.order`: Crear una clase que herede de `sale.order` y agregar el campo `modo_pago`.
3. Configurar el campo: Definir el campo como un `selection` con opciones: metálico, transferencia bancaria, tarjeta.
4. Heredar la vista: Modificar la vista existente para mostrar el nuevo campo.

HEREDAR LA VISTA

→ Objetivo: Modificar una vista existente para incluir nuevos campos.

→ Pasos:

1. Usar la herramienta de desarrollador en Odoo para obtener el ID externo de la vista.
2. En la carpeta `views`, crear un archivo XML para la vista heredada.
3. Usar el ID externo para heredar la vista y agregar el nuevo campo.
4. Indicar en qué parte de la vista se debe mostrar el nuevo campo.

INSTALACIÓN DEL MÓDULO

1. Actualizar la lista de aplicaciones.
2. Instalar el módulo.
3. Comprobar que el nuevo campo "modo de pago" aparece en el módulo de Ventas.

WUOLAH

EL LOG DE ODOO

- Configurar el log → en el archivo `odoo.conf`, definir el archivo de log y el nivel de detalle (ej: `:DEBUG`).
- Ver el log → los logs se pueden ver en el archivo `odoo.log` o en la consola de PyCharm.
- Analizar errores → usar los logs para identificar y corregir errores en el código.

“ANALIZANDO EL LOG A EJECUTAR ODOO DESDE PYCHARM” (CASO PRÁCTICO)

- Nos aparece en el log al ejecutar Odoo desde PyCharm.
- Podemos cambiar el nivel de log a `:DEBUG` para obtener más información.

ERRORES O DUMPS

- Tipos de errores:
 - Errores de sintaxis → PyCharm resalta errores de sintaxis en tiempo real (ej: etiquetas XML mal cerradas).
 - Errores en tiempo de ejecución → Odoo muestra errores al instalar o ejecutar el módulo (ej: ID externo incorrecto).
- Podemos usar la información proporcionada por los errores para corregir el código.

Tema 15: Desarrollo de nuevos componentes II: Administración e informes

AGREGAR CAMPOS EN FORMULARIOS

→ Para añadir nuevos campos a formularios de Odoo desde la interfaz:

1. Activar el modo desarrollador.
2. Identificar el modelo del formulario viendo sus metadatos y se añade el campo nuevo en la base de datos.
3. Modificar la vista del formulario para mostrar el campo en la posición deseada.

→ Estos cambios son temporales y pueden perderse al actualizar Odoo.

“EXTRACCIÓN DE DATOS” (CASO PRÁCTICO)

→ Objetivo → extraer un listado de productos en venta en formato Excel. Pasos:

1. Ir al módulo de Ventas y acceder a productos.
2. Abrir vista de lista.
3. Seleccionar todos los productos.
4. Usar acción/exportar.
5. Seleccionar campos deseados y formato de salida.
6. Pulsar "Exportar".

ADMINISTRADOR DE USUARIOS

→ Los usuarios en Odoo se gestionan desde el módulo de ajustes.

- Para crear un usuario nuevo se necesita su nombre, email y foto.
- La eliminación se hace seleccionando el usuario y usando acción/suprimir.
- Los permisos se pueden configurar por módulo y tipo de usuario (interno, portal o público).
- El sistema de grupos permite asignar conjuntos de permisos a múltiples usuarios a la vez.

→ Diferentes secciones en la sección “Permisos de acceso”:

- Multicompañía → si un usuario está en más de una compañía.
- Tipo de Usuario:
 - “usuario interno” → trabajará internamente con Odoo.
 - “Portal” → tiene reglas preestablecidas, no se pueden gestionar permisos.
 - “Público” → tiene reglas preestablecidas, no se pueden gestionar permisos.
- <Módulos> → aparecen los diferentes módulos y los niveles de permisos a configurar dentro de ellos.
- Configuración técnica, permisos extra y otros → podemos marcar/desmarcar permisos extra para ofrecer más o menos privilegios al usuario.

CREAR UN INFORME CON QWEB

→ QWeb → el motor de plantillas XML de Odoo usado para generar fragmentos y páginas HTML.

→ Para crear un informe nuevo, hay que identificar el modelo donde aparecerá, crear una vista XML con la etiqueta 'report' especificando que es tipo QWeb, y definir el contenido en una etiqueta 'template'.

→ El archivo XML debe incluirse en el *manifest* del módulo.

Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato
→ Planes pro: más coins

pierdo
espacio



Necesito
concentración

ali ali ooh
esto con 1 coin me
lo quito yo...

WUOLAH

“INTERNAL SERVER ERROR” (CASO PRÁCTICO)

→ Objetivo: Resolver el error que impide acceder a Odoo.

→ Pasos:

1. Abrir administrador de tareas.
2. Ir a pestaña servicios.
3. Localizar odoo-server-13.0.
4. Abrir servicios.
5. Reiniciar el servicio de Odoo.

ERRORES COMUNES

→ Tres errores frecuentes al desarrollar en Odoo:

1. No crear el archivo *init.py* para importar los modelos Python, lo que impide que se creen las tablas en PostgreSQL.
2. Olvidar incluir las nuevas vistas XML en la sección 'data' del *manifest*, lo que hace que no se carguen.
3. No indicar en el *manifest* las dependencias de otros módulos necesarios para el funcionamiento del módulo actual → sección 'depends'.

WUOLAH

CRITERIOS

Tema 10

3a) Adapta sistemas ERP-CRM identificando los requerimientos de un supuesto empresarial y utilizando las herramientas proporcionadas por los mismos.

4a) Se han identificado las posibilidades de adaptación del ERP-CRM.

Tema 11

2h) Se han documentado las operaciones realizadas y las incidencias.

5a) Se han reconocido las sentencias del lenguaje propio del sistema ERP-CRM.

Tema 12

5a) Se han reconocido las sentencias del lenguaje propio del sistema ERP-CRM.

Tema 14

5e) Se ha verificado el correcto funcionamiento de los componentes creados.

5f) Se han documentado todos los componentes creados o modificados.

Tema 15

5e) Se ha verificado el correcto funcionamiento de los componentes creados.

5f) Se han documentado todos los componentes creados o modificados.