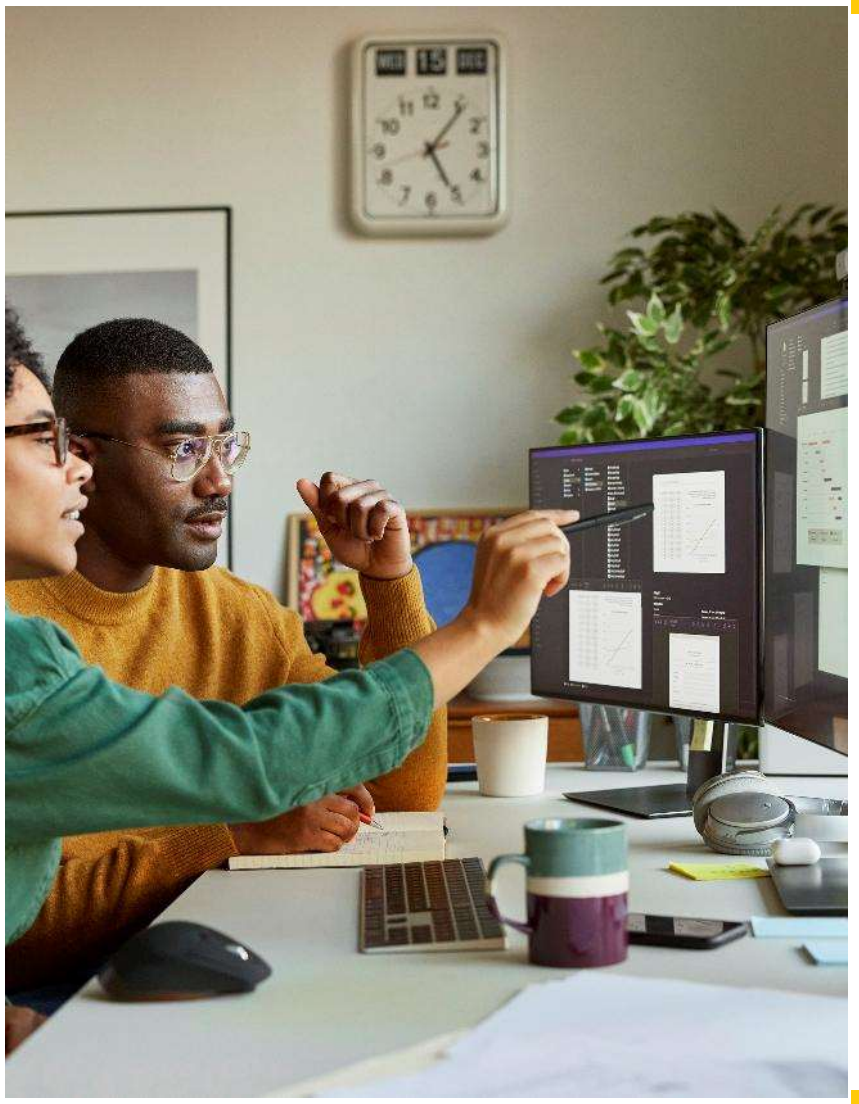
Sistemas de Gestión Empresarial

Tema 3
Salesforce en entorno desarrollador

llanos.soro@educa.madrid.org

Objetivos del tema

1. Desarrollador Salesforce
2. Modelo de datos: objetos estándar vs. personalizados
3. Relaciones entre objetos
4. Seguridad: perfiles, roles, permisos y sharing rules



Principios de un desarrollador Salesforce

Los principios de un desarrollador Salesforce incluyen el dominio de las tecnologías de la plataforma (como **Apex y JavaScript),** la capacidad de integración con sistemas externos a través de APIs, y la creación de soluciones personalizadas que extiendan la funcionalidad del CRM.

También es fundamental alinearse con los valores centrales de Salesforce (confianza, innovación, éxito del cliente, sostenibilidad y equidad), pensar de forma arquitectónica, utilizar herramientas tanto de código como sin código, y enfocarse en el **análisis de requisitos y el diseño de soluciones efectivas**.

Principios de un desarrollador Salesforce

Responsabilidades clave

- **Análisis de requisitos:** Recolectar y comprender las necesidades específicas de los usuarios y las empresas.
- **Diseño de soluciones:** Crear planes para aplicaciones personalizadas e integraciones en la plataforma Salesforce.
- **Desarrollo y pruebas:** Escribir código en lenguajes como Apex y JavaScript para construir y probar aplicaciones de software.
- **Personalización de aplicaciones:** Usar herramientas como *Lightning App Builder* para configurar y personalizar interfaces de usuario.
- **Integración de sistemas:** Conectar Salesforce con otros servicios web y sistemas mediante APIs.
- **Mantenimiento y actualización:** Asegurar que las aplicaciones sean seguras, escalables y estén actualizadas con las últimas funciones de Salesforce.

Principios de un desarrollador Salesforce

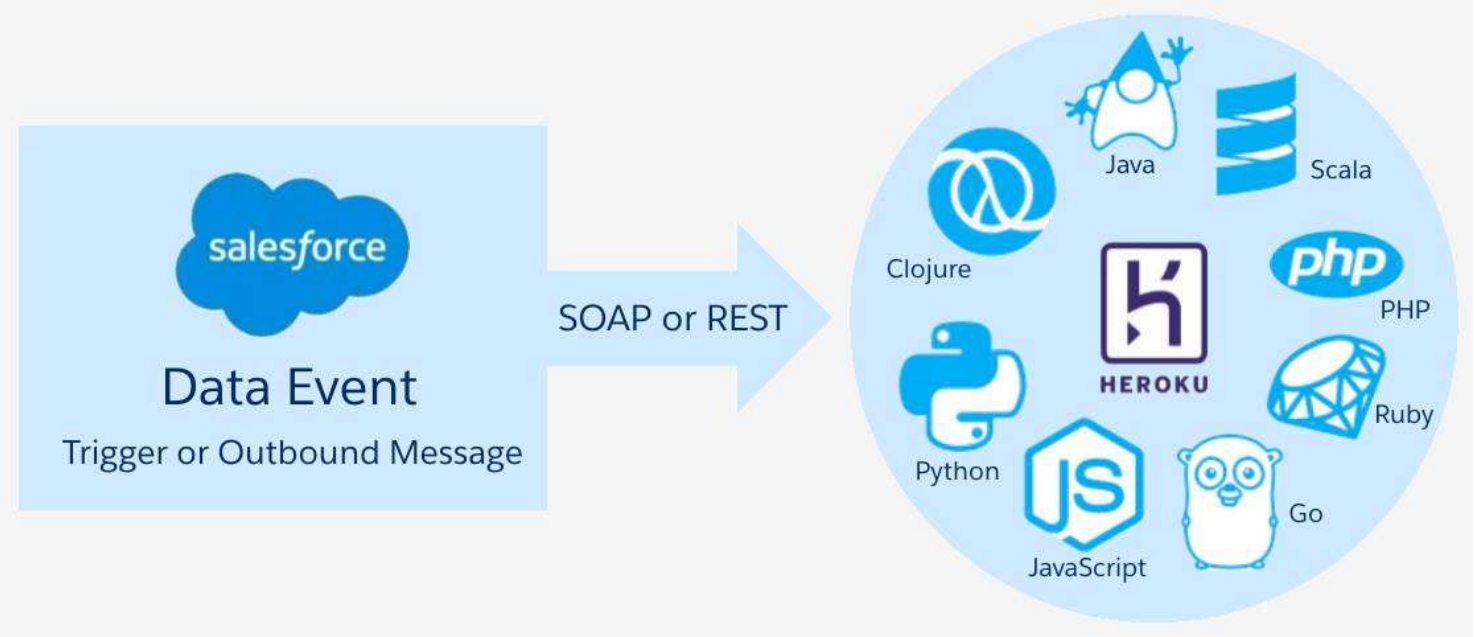
Habilidades principales

- **Lenguajes específicos de Salesforce:** Dominio de Apex, el lenguaje de programación propietario de Salesforce.
- **Lenguajes de programación:** Conocimientos de Java y JavaScript para añadir elementos interactivos y desarrollar más funcionalidades en las aplicaciones.
- **Desarrollo frontend:** Experiencia **con Lightning Component Framework** y CSS para diseñar y estructurar páginas web.
- **Desarrollo backend:** Comprensión de consultas SOQL (**Salesforce Object Query Language**), operaciones DML y *triggers*.
- **Modelado de datos:** Capacidad para crear y optimizar objetos personalizados que almacenen la información de la empresa de manera eficiente.
- **Resolución de problemas:** Fuertes habilidades analíticas y capacidad para resolver problemas de forma creativa.

2. Entorno de programación Salesforce

Salesforce es una **plataforma PaaS (Platform as a Service)** llamada **Salesforce Platform**, diseñada para desarrollar aplicaciones empresariales en la nube.

Su entorno de programación combina herramientas de **configuración (low-code)** y **desarrollo avanzado (pro-code)**.



2. Entorno de programación Salesforce

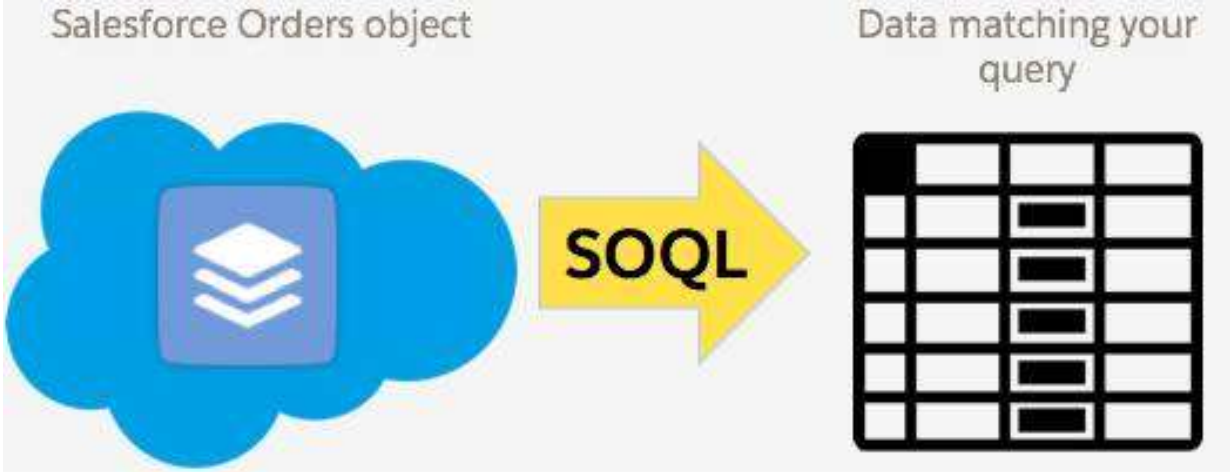
Apex es el lenguaje de programación propietario de Salesforce, **similar a Java**, fuertemente tipado, orientado a objetos y ejecutado en el servidor.

- Sintaxis similar a Java.
- Se ejecuta en los servidores de Salesforce (no en el cliente).
- Se usa para controlar la lógica de negocio, realizar operaciones con datos (DML) y automatizar procesos.
- Tiene límites de ejecución para garantizar rendimiento y seguridad en la nube .

2. Entorno de programación Salesforce

Salesforce tiene sus propios lenguajes de consulta de datos:

•**SOQL (Salesforce Object Query Language):** similar a SQL, se usa para recuperar registros de objetos.
•**SOSL (Salesforce Object Search Language):** usado para búsquedas de texto en varios objetos al mismo tiempo.



2. Entorno de programación Salesforce

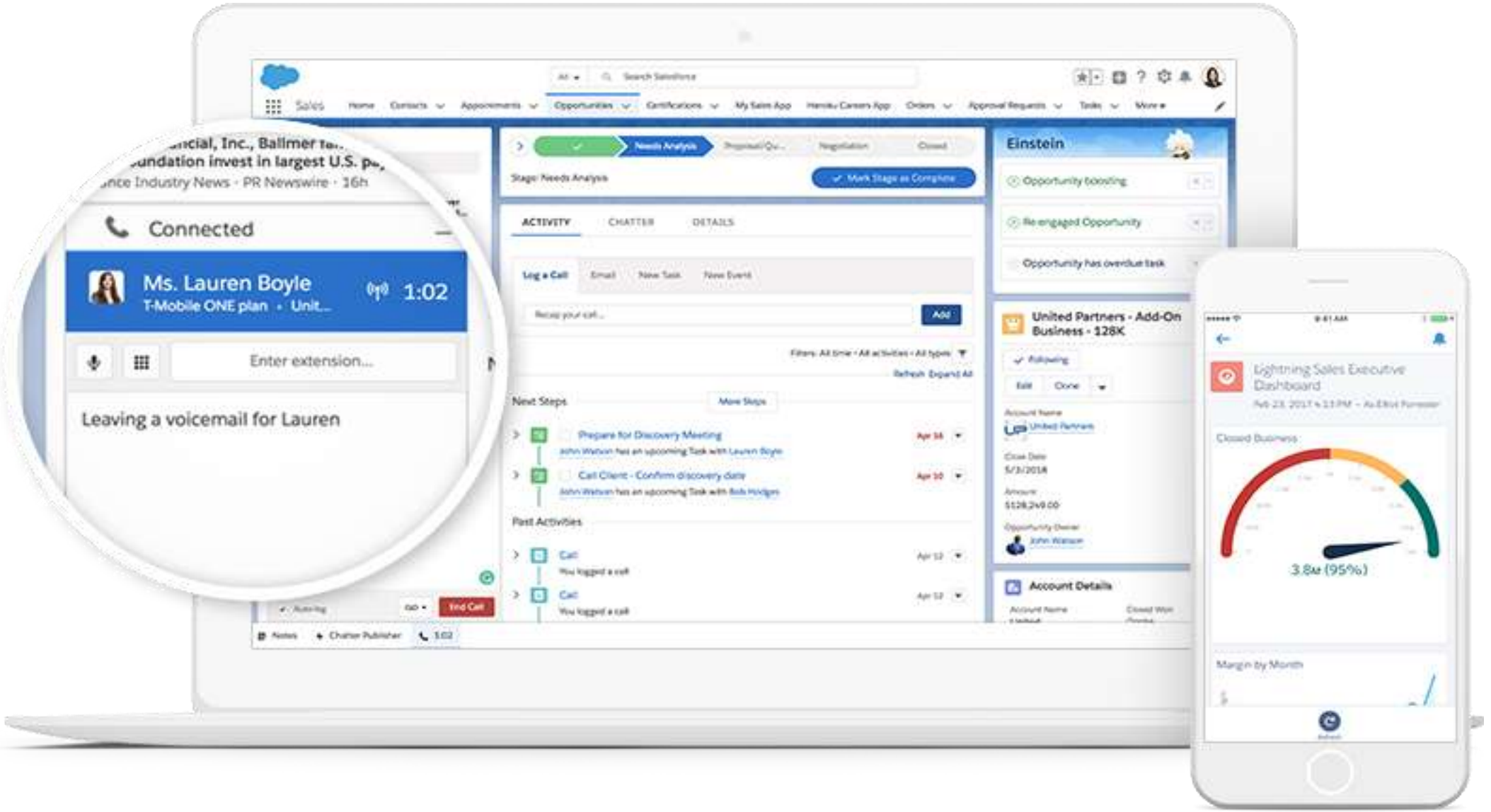
Visualforce:

Un framework más antiguo basado en etiquetas HTML-like, usado para crear interfaces personalizadas con controladores Apex.

Lightning (actual estándar):
Framework moderno basado en componentes web.
Incluye **Lightning Web Components (LWC)**, desarrollados con **JavaScript moderno, HTML y CSS**.
Permite experiencias rápidas, modulares y responsivas.

2. Entorno de programación Salesforce

2. Entorno de programación Salesforce

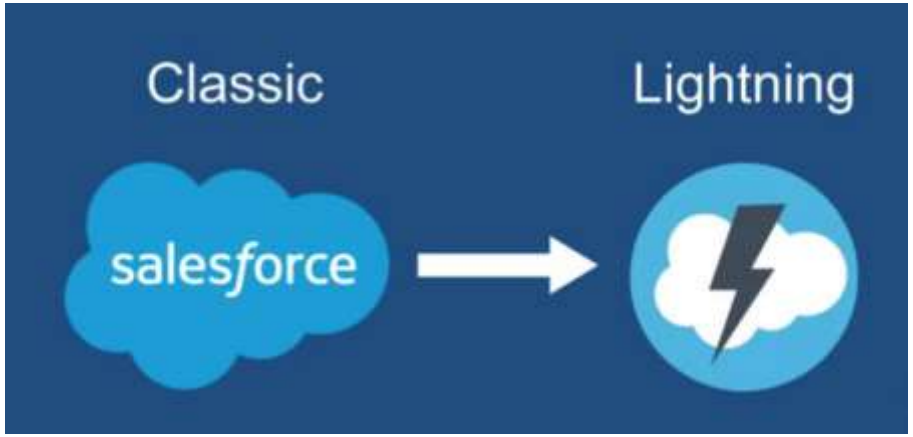


2. Entorno de programación Salesforce

Diferencias entre Salesforce Classic y Lightning Experience

Salesforce ha evolucionado desde su interfaz original (Salesforce Classic) hasta su versión moderna (Lightning Experience).
Aunque ambas comparten la misma base de datos y funcionalidades principales, Lightning ofrece una experiencia más visual, eficiente y adaptable para usuarios y administradores.

Classic aún está disponible, pero Salesforce impulsa el uso exclusivo de Lightning Experience, ya que todas las innovaciones actuales se centran en ella.

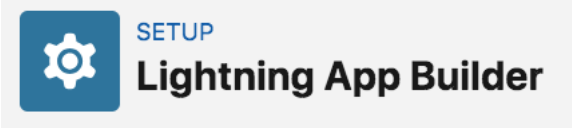


2. Entorno de programación Salesforce

Característica	Salesforce Classic	Lightning Experience
Interfaz de usuario	Basada en menús y enlaces de texto. Diseño más antiguo.	Moderna, intuitiva, con componentes visuales y paneles dinámicos.
Velocidad de navegación	Lineal (un registro a la vez).	Navegación rápida y adaptable con pestañas, subpestañas y componentes.
Personalización	Limitada a page layouts y campos.	Amplia: se pueden añadir componentes Lightning, apps, flujos y dashboards personalizados.
Automatización y desarrollo	Requiere más programación (Apex, Visualforce).	Potente suite declarativa: Flow Builder, App Builder, Dynamic Forms, etc.
Informes y paneles	Menos visuales, estáticos.	Dashboards interactivos, filtros dinámicos, gráficos modernos.

3. Principales funciones de Salesforce Lightning

- Lightning App Builder : Diseña y personaliza páginas sin código (inicio, registro, app).
- App Launcher :Accede a todas las apps y objetos desde un solo menú.
- Lightning Record Pages : Personaliza cómo se muestran los registros (con Dynamic Forms).
- Flows (Flow Builder) : Automatiza procesos de negocio sin programación.
- Informes y Dashboards Lightning : Analiza datos con paneles visuales y dinámicos.



3. Principales funciones de Salesforce Lightning

•Dynamic Forms & Actions : Muestra campos o botones según condiciones o perfil.

•App Manager : Crea y administra aplicaciones Lightning (iconos, navegación, permisos).

•Quick Actions & Global Actions : Realiza tareas o crea registros rápidamente.

•List Views & Kanban : Visualiza y gestiona registros de forma interactiva.

•Experiencia móvil integrada : Usa Salesforce desde la app móvil con la misma interfaz.

4. Modelo de datos: objetos estándar

Son los que Salesforce incluye por defecto y cubren procesos comunes de negocio.

•Cuenta (Account): representa una empresa o cliente.

•Contacto (Contact): persona asociada a una cuenta.

•Oportunidad (Opportunity): posible venta o negocio.

•Caso (Case): solicitud de soporte o incidencia.

•Usuario (User): representa a un empleado o persona con acceso al sistema.

4. Modelo de datos: objetos estándar

Características:

No se pueden eliminar, pero sí personalizar (añadir campos, cambiar etiquetas, etc.).

Ya incluyen relaciones predefinidas y permisos configurados.

4. Modelo de datos: objetos personalizados

Son creados por el administrador o desarrollador para adaptar Salesforce a las necesidades específicas del negocio.

Ejemplo:
Una empresa de construcción podría crear un objeto Proyecto para controlar obras, con campos como:

•Nombre del Proyecto

•Presupuesto

•Fecha de inicio

•Estado

4. Modelo de datos: objetos personalizados

Ventajas:

- Totalmente configurables.
- Pueden relacionarse con objetos estándar.
- Se pueden usar en informes, Flows, y procesos automatizados.

5. Seguridad

Salesforce implementa seguridad en múltiples niveles:

a) Seguridad a nivel de objeto

Controla quién puede crear, leer, editar o eliminar un tipo de registro.

Configurada mediante Perfiles o Permisos.

b**) Seguridad a nivel de campo**

Controla si un usuario puede ver o editar campos específicos.

Ejemplo: ocultar el campo “Salario” a usuarios del perfil “Ventas”.

5. Seguridad

c) Seguridad a nivel de registro

Controla qué registros específicos puede ver un usuario.

Roles: definen jerarquías (por ejemplo, un gerente ve los registros de su equipo).

Sharing Rules: permiten compartir registros con grupos o roles específicos.

Manual Sharing: permite compartir registros individuales.

Ejemplo práctico:
Un usuario de ventas solo puede ver sus propias oportunidades, pero el gerente de ventas puede ver las de todo el equipo.

5. Seguridad


Organizational level security: IP ranges, login hours
Object level security: profiles
Record level security: role hierarchy
Field level security: field access (read/write)

6. Automatización: validation rules, Flows, Process Builder

Salesforce ofrece potentes herramientas sin código (declarativas) para automatizar procesos.

Validation Rules (Reglas de validación)

Garantizan la integridad de los datos.

Usan fórmulas booleanas: si el resultado es TRUE, se muestra un mensaje de error.

6. Automatización: validation rules, Flows, Process Builder

Process Builder

Permite definir condiciones y acciones automáticas.

Ejemplo: cuando unà oportunidad cambia a “Cerrada - Ganada”, crear automáticamente una tarea de seguimiento.

Actualmente, Salesforce recomienda migrar procesos a Flows, ya que Process Builder quedará en desuso.

6. Automatización: validation rules, Flows, Process Builder

Flows (Flow Builder)

La herramienta más potente de automatización visual.
Tipos principales:

•Screen Flows: incluyen pantallas para interacción del usuario.

•Autolaunched Flows: se ejecutan automáticamente (por cambio de registro, programación o evento).

Ejemplo :
Un Flow que, al crear un nuevo Proyecto:

Asigne automáticamente un responsable.

Envíe una notificación al gerente.

6. Automatización: validation rules, Flows, Process Builder