Salesforce si basa su un modello MVC.
MVC è l'acronimo di **Model View Controller** ed è un modello di architettura software che suddivide la stessa in tre tipi di componenti, in modo da separare la rappresentazione delle informazioni dall'interfaccia utente.
Un **controller** invia comandi alla vista(View) associata, per modificare la presentazione della vista(View) del modello .
Può anche inviare comandi al modello per aggiornare lo stato del modello .
I Controller sono scritti in codice Apex e finiscono per controllare e applicare tutta la logica aziendale.
Le pagine interagiscono con il controller attraverso i componenti che trasferiscono i dati e specificano cosa succede quando l'utente interagisce effettivamente con l'interfaccia utente.
Salesforce ha controller predefiniti per molte delle azioni standard come Visualizza, Modifica, Salva. Se si desidera aggiungere un nuovo comportamento, è possibile estendere o creare nuovi controller (controller personalizzati) in Apex.

Un **Model** notifica alle viste(View) e ai controller associati quando si è verificato un cambiamento nel suo stato.
Questa notifica consente alle viste(View) di produrre output aggiornati e ai controller di modificare il set di comandi disponibile.
I Model sono oggetti del database in Salesforce.
Include gli oggetti Salesforce standard come Lead, Contact, Account,Opportunity ecc.
Ma anche oggetto personalizzati.
Una V**iew** richiede al Modello le informazioni necessarie per generare una rappresentazione di output.
È possibile scrivere le proprie pagine(le V**iew)** usando Visualforce(VF).
Visualforce è il framework di interfaccia utente basato su componenti per la piattaforma Force.com.
Include un linguaggio di markup basato su **tag, simile all'HTML.**
Con Visualforce è possibile :

- Sostituire le pagine Salesforce standard con la propria pagina Visualforce.
- Creare il proprio controllo di flusso personalizzato per un'applicazione.

- -Definire schemi di navigazione e regole specifiche per i dati, per ottimizzare l'interazione efficace con le applicazioni.
- -Può essere integrato con HTML, CSS, Ajax, jQuery.

Visualforce adotta lo stile di sviluppo **Model View Controller (MVC).**
Ogni pagina Visualforce è associata a un controller.
È possibile fare uso di controller standard già creati oppure creare un controller usando il linguaggio Apex.
View: è la presentazione dei dati, quindi l'interfaccia utente, si hanno **Pages e Components.**

- **Pages:** Sono i mattoni dell'interfaccia utente .Visualforce utilizza HTML per definire l'aspetto dell'interfaccia dell'applicazione. **Ogni pagina** è referenziata da un URL unico, proprio come una normale pagina web, e contiene componenti Visualforce che possono essere richiamati da semplici tag all'interno della pagina.
- **Components:** esistono componenti Visualforce standard e personalizzati.

E possibile riutilizzarli su più pagine.
Possono anche essere abbinati a CSS.

Domain

Per poter operare pienamente con gli strumenti di sviluppo
della UI messi a disposizione da Salesforce occorre creare
preventivamente un dominio personalizzato.

Visualforce Page tags

<apex:page>

Il tag di base che **crea una pagina Visualforce**. Può essere utilizzato solo una volta in una pagina e tutti i tag visualforce devono essere racchiusi in questo tag.

<apex:pageblock>

Un tag che crea un'area all'interno della pagina in cui è possibile creare più sezioni e possono essere visualizzati campi, pulsanti, tabelle o collegamenti.
Di default eredita lo stile di pagina standard di Salesforce.

<apex:pageBlockSection>

Questo tag creare una sezione all'interno di un blocco di pagina *<apex:pageblock>*.

È possibile creare più sezioni in un blocco di pagina e ciascuna sezione può essere utilizzata per visualizzare qualsiasi campo (input / output).
Un componente pageBlockSection è costituito da una o più colonne, per impostazione predefinita è uguale a due, ognuna delle quali si estende su due celle: una per l'etichetta di un campo e una per il suo valore.
Ogni componente trovato nel corpo di un <apex:pageBlockSection> viene inserito nella cella successiva libera di una riga finché non viene raggiunto il numero di colonne.

<apex:pageBlockButtons>

E un contenitore che contiene Tag <apex: commandbutton> e serve a genere bottoni sopra e sotto ad una section in stile Saleforce

<apex:commandButton>

Tag per cerare un input di tipo bottone col stile si Salesforce

<apex:form>

Tag per cerare un form col stile si Salesforce

<apex:pageMessage>

Questo componente deve essere utilizzato per presentare messaggi personalizzati nella pagina utilizzando il modello di Salesforce per errori, avvisi o altri tipi di messaggi per una determinata gravità.
Si deve adopera l'attributo obbligatorio Severity che determina la gravità del messaggio i valori supportati sono: **'confirm', 'info', 'warning', 'error'.**
L' attributo Strength (non è obbligatorio) determina la visibilità e le dimensioni dell'icona visualizzata accanto al messaggio. Con il valore 0 non viene inserita nessuna immagine con 1 fino al 3 cambia l'icona

<apex:panelBar>

Un'area della pagina che include uno o più tag <apex:panelBarItem> essa possono estendersi quando un utente fa clic sull'intestazione associata.
Quando un <apex: panelBarItem> viene esteso,l'intestazione e il contenuto dell'elemento vengono visualizzati mentre il contenuto di tutti gli altri elementi è nascosto. Se un altro <apex: panelBarItem> viene aperto, il contenuto dell'elemento che veniva visualizzato viene nascosto.

Un <apex: panelBar> può includere fino a 1000 tag <apex: panelBarItem>.

<apex:panelBarItem>

E un elemento che si puè utilizzare solo all'interno di un Tag <apex: panelBar> .
I suoi attibbuti onLeave e onEnter consentono di gestire gli eventi di quando un panelBarItem viene aperto o chiuso.

<apex:panelGrid>

Genera elemento tabella HTML in cui ogni componente figlio viene inserito in una cella corrispondente nella prima riga fino al raggiungimento del numero di colonne.
A quel punto, il componente successivo si posiziona nella prima cella libera della riga successiva .

<apex:tabPanel>

Un'area della pagina visualizzata come un insieme di schede.
Quando un utente fa click su una scheda, viene visualizzato il contenuto associato alla scheda <apex:tab>,nascondendo il contenuto delle schede.
L'attributo switchType determina il metodo per passare da una scheda all'altra.
I valori possibili sono client, server e ajax.

(Se non specificato, questo valore viene impostato su server).

<apex:tab>

Una singola scheda in un <apex: tabPanel>.
Il componente <apex: tab> deve essere contenuto in un <apex: tabPanel>.

<apex:toolbar>

Una barra degli strumenti orizzontale stilizzata che può contenere un qualsiasi numero di componenti figlio.
Per impostazione predefinita, tutti i componenti figlio sono allineati al lato sinistro della barra degli strumenti.

<apex:toolbarGroup>

Raggruppa un gruppo di componenti, nella barra degli strumenti, può essere allineata a sinistra o a destra .
Il componente <apex: toolbarGroup> deve essere un componente figlio di <apex:toolbar>.
I suoi attributi sono :

- ItemSeparator(API version 10.0) utilizzato per separare i componenti della barra degli strumenti.I valori possibili sono none,line,square,disc,grid .

Se non specificato, questo valore viene impostato su none.

- Id: un identificatore che consente al componente della barra degli strumenti di fare riferimento ad altri componenti nella pagina.

Input tags

I tag di input vengono utilizzati per acquisire l'input dell'utente, per qualsiasi Fild di un oggetto standard/personalizzato, e rispetta tutti i requisiti impostati sul Fild, ad esempio se il campo è obbligatorio, o univoco, o se l'utente corrente dispone dell'autorizzazione per visualizzarlo o modificarlo.

<apex:inputText>

E un input HTML di tipo text.
Serve per ottenere l'input dell'utente per un metodo di un controller che non corrisponde a un Fild di un oggetto Salesforce.
Questo componente non utilizza lo stile di Salesforce.
Poiché non corrisponde a un qualsiasi dato/Fild di un oggetto, è richiesto un codice personalizzato per utilizzarlo il valore immesso dall'utente.

<apex:inputSecret>

Un input HTML di tipo password.
Serve per ottenere l'input (valore nascosto visivamente)dell'utente per un metodo controller che non corrisponde a un Fild di un oggetto Salesforce.

<apex:inputTextarea>

Un input di tipo TextArea.
Serve per ottenere l'input dell'utente per un metodo controller che non corrisponde a un Fild di nessun un oggetto Salesforce.

<apex:inputCheckbox>

Un input HTML di tipo checkbox.
Serve per ottenere l'input dell'utente per un metodo controller che non corrisponde a un Fild di un oggetto Salesforce.

Output tags

I tag di output vengono utilizzati per visualizzare e stampare il testo sullo schermo.

<apex:outputLabel>

Una Label per un campo di input o di output.
Serve per fornire un'etichetta per un metodo di un controller che non corrisponde ad un Fild si nessun oggetto Salesforce.

<apex:outputLink>

Questo tag crea un collegamento a un URL. Quando si fa clic su questo collegamento, viene aperto l'URL specificato nell'attributo value.

Il corpo di un <apex: outputLink> può essere un testo o un'immagine.