



## Salesforce Project Ready

Entry Level Training & Professional Development  
Program

**Project Title : Online Training Management**

## Day 1 :

Topic : Brief explanation of Salesforce and CRM, Comparison of Excel data and Salesforce

Milestone / Activities: Difference between Standard Objects and Custom Objects

Detailed Description : Salesforce is a Cloud based software company, that gives platform to businesses to relate their Marketing, IT, Sales, Commerce, and Services to their customers. Salesforce has web based Software. Salesforce Integrated into many categories to improve their business. Now a days salesforce is using on Facebook, LinkedIn, youtube etc.



It is a No 1 CRM platform, i.e, Customer Relationship Management which makes direct interaction between customer and business by apps. Here customers can make direct marketing by given platform and also track customers marketing strategy.

Application: In salesforce we can create number of application by using many platform like Lightning Components, Apex, VisualForce. By comparing this to Excel we say that application is workbook.

Object : It organises the data in application. Salesforce already given database that includes few Objects and fields that helps customers/clients to run better those objects are called Standard Objects, and which we create by requirements are called Custom Objects.

Salesforce Developer Edition can be created by two ways

They are:

- i) Trailhead
- ii) salesforce web site

Milestone Activities Screenshot:

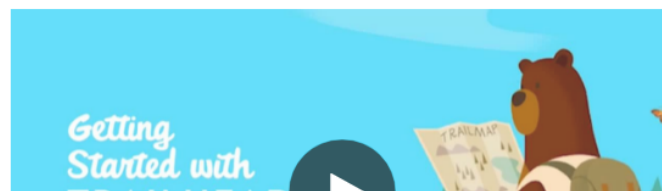
Login to the salesforce



## WELCOME, KADAPA

This is your Trailhead Playground, where you can complete hands-on challenges for Trailhead, and try out new Salesforce features and customizations.

Just getting started? Watch this quick video to learn tips for success.



## Day 2 :

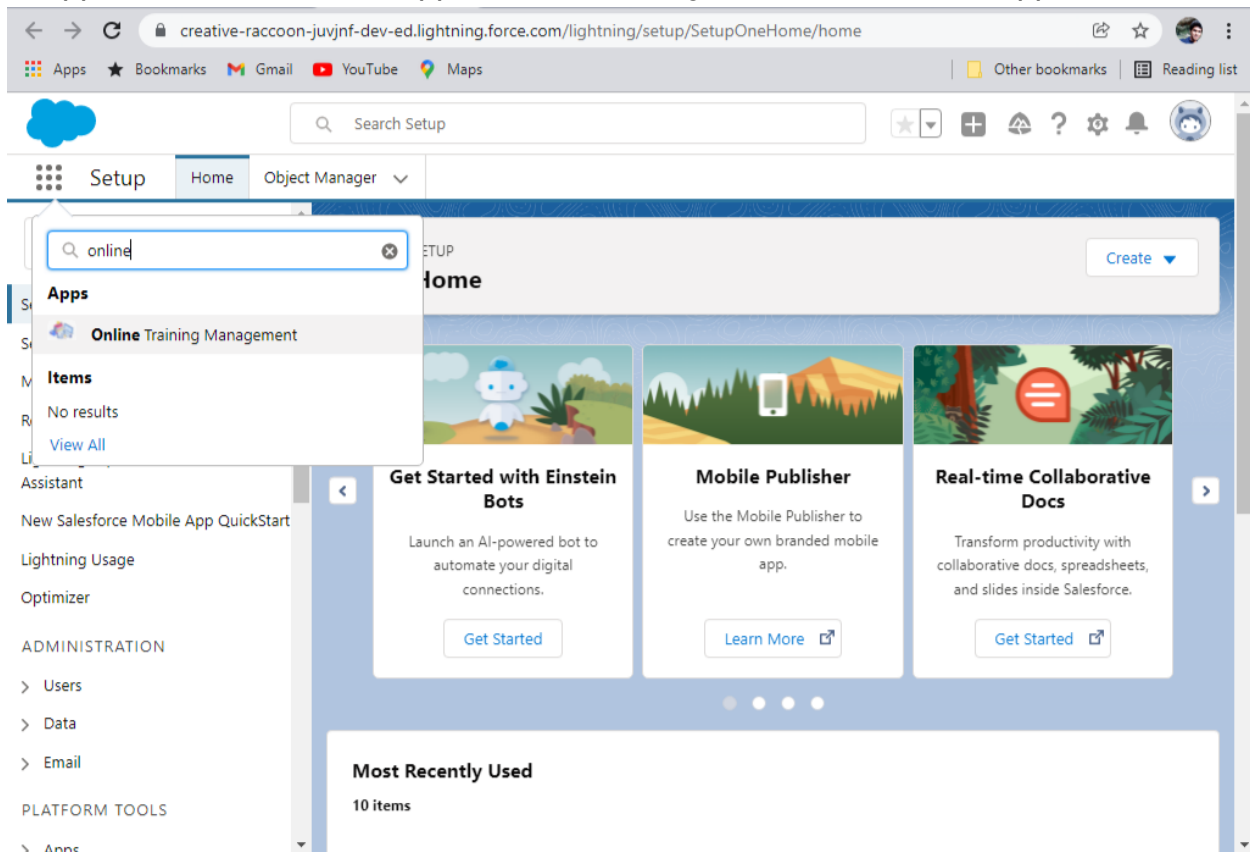
Topic : Creating Application, Objects, Fields, and Validation rules

Milestone / Activities: Created validation rules

Detailed Description: Project aim is to create App for "**Online Training Management**".

This app is created by Lightning App. In available tabs, add Home tab.

In App launcher we find our application, following is the screenshot of app



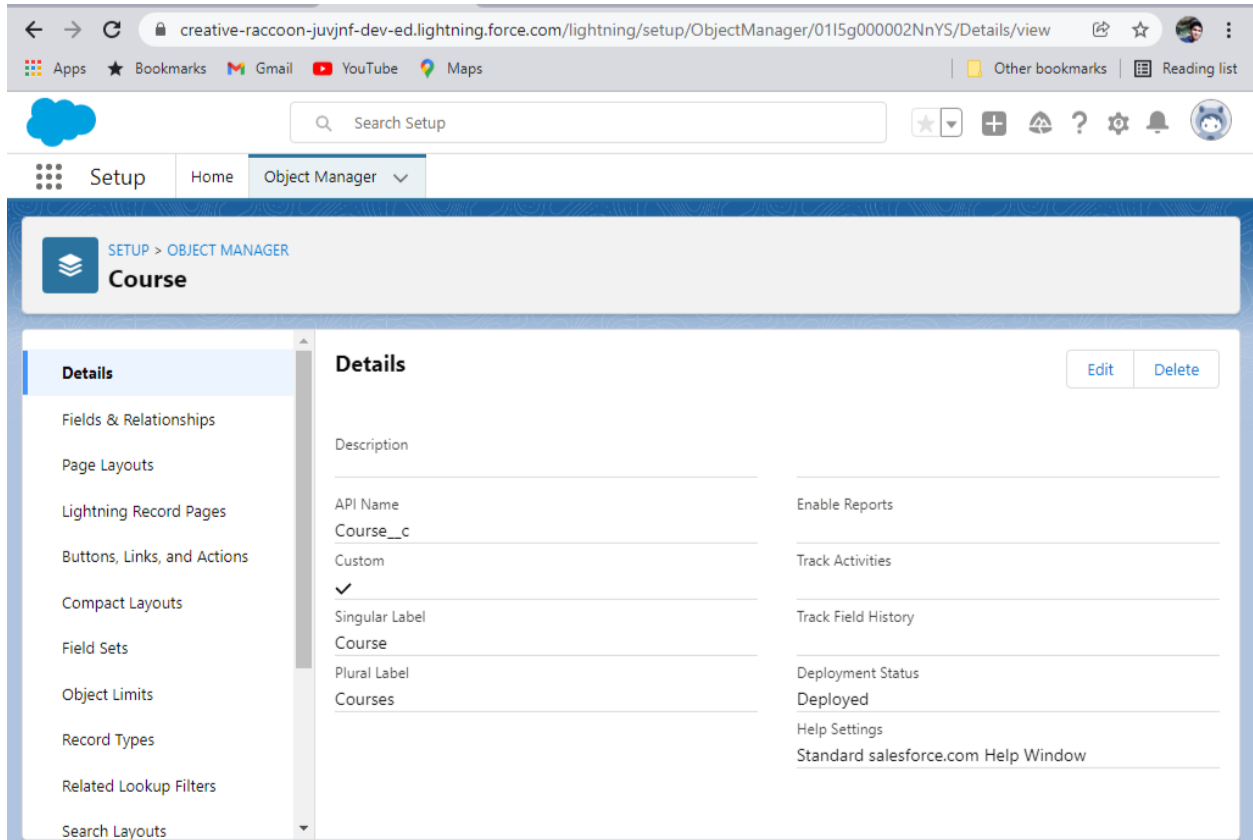
Create custom Objects either using Schema Builder or Custom Object. The following custom objects are created for the Online training Management are

### i) Courses

- Rename Record Name : Course ID
- Data type : Auto Number
- Display Format : C-{000000}
- Starting Number : 1
- Deployment status : Deployed
- Allow Search : Access
- Add Notes and Attachment : Selected

- Add to Navigation Tab : Selected

### Snapshot of Course Custom Object :



### Custom Course Object Custom Fields are

- Course Name as label and with datatype Text.
- Course Duration(In Months) as label with datatype picklist by entering values are(1,2,3,4,5,6)
- Course Category as Label with picklist datatype and values are (Cloud Computing, Data Management, Cyber Security)
- Course Fee as label with Currency datatype with length(5,4).
- Course Description as label with Text datatype and length(50)

### Validation Rules for Course Object is

- Course Fee should not be negative value and should be less than \$3000.

## Snapshot of Course Custom Object Custom Fields:

The screenshot shows a web browser window with the URL `creative-raccoon-juvjnf-dev-ed.lightning.force.com/lightning/o/Course_c/new?count=1&nooverride=1&useRecordT...`. The browser's address bar and tabs are visible. The main content area displays the 'Information' modal for creating a new Course record. The modal has a title bar 'Information' and a close button. The fields are as follows:

- Course ID**: A text field.
- Owner**: A dropdown menu showing 'Kadapa Charan'.
- \* Course Name**: A required text field.
- \* Course Duration(In Months)**: A required dropdown menu with '--None--' selected.
- \* Course Category**: A required dropdown menu with '--None--' selected.
- \* Course Fee**: A required text field.
- Course Description**: A text area.

At the bottom of the modal, there are three buttons: 'Cancel', 'Save & New', and 'Save'.

## ii) Trainer

- Rename Record Name : Trainer ID
- Data type : Auto Number
- Display Format : T-{000000}
- Starting Number : 1
- Deployment status : Deployed
- Allow Search : Access
- Add Notes and Attachment : Selected
- Add to Navigation Tab : Selected

Trainer Object Custom Fields are

- Full Name as label with Text datatype contains length(25).
- Email as both label and datatype and it should not take duplicate mails.
- Mobile as label with Phone datatype with length(13).
- Whatsapp as label with Phone datatype with length(13).
- Date of Birth as label with date datatype.
- Age as label using formula as datatype and return type as Number.
- Experience as label using picklist datatype and values are(0,1,2,3,4)
- LinkedIn Profile as label using URL datatype.

- Expertise as label using text datatype.

Validation Rules for Trainer Object are

- Mobile and Whatsapp should be minimum of 10 digits and maximum of 15 digits.
- Full Name, Phone, Date of Birth cannot be empty.
- Trainer age cannot be less than 25 years.

Snapshot of Trainers Custom Object and its Fields:

The screenshot shows a web browser window with the URL `creative-raccoon-juvjnf-dev-ed.lightning.force.com/lightning/o/Trainer__c/new?count=2&nooverride=1&useRecordTy...`. The browser's address bar and tabs are visible. The main content area shows a Salesforce Lightning console with a sidebar on the left containing 'Trainers' and 'Recently' sections. The main panel displays a 'New Trainer' form. The form has a title bar 'New Trainer' and a tab 'Information'. The form fields are arranged in two columns. The left column contains 'Trainer ID', '\* Full Name' (with a red asterisk), '\* Email' (with a red asterisk), '\* Date Of Birth' (with a red asterisk and a calendar icon), and 'Experience' (a dropdown menu with '--None--' selected). The right column contains 'Owner' (a user icon and the name 'Kadapa Charan'), 'Expertise', 'LinkedIn', 'Phone', and 'Whatsapp Number'. At the bottom of the form are three buttons: 'Cancel', 'Save & New', and 'Save'.

### iii) Students

- Rename Record Name : Student ID
- Data type : Auto Number
- Display Format : S-{000000}
- Starting Number : 1
- Deployment status : Deployed
- Allow Search : Access
- Add Notes and Attachment : Selected
- Add to Navigation Tab : Selected

Custom Fields for this Object are

- First Name as label with Text datatype contains length(25).
- Last Name as label with Text datatype contains length(25).

- Email as both label and datatype and it should not take duplicate mails.
- Mobile as label with Phone datatype with length(13).
- Whatsapp as label with Phone datatype with length(13).
- Date of Birth as label with date datatype.
- Enrolment Date as label using Auto Generated(date) and it should be locked/read only field.

Validation rules for Student Object are

- Mobile and Whatsapp should be minimum of 10 digits and maximum of 15 digits.
- Last Name, Phone, Date of Birth cannot be empty.
- Trainer age cannot be less than 16 years

Snapshot of Student Customs and its Fields:

The screenshot shows a web browser window with the URL `creative-raccoon-juvjnf-dev-ed.lightning.force.com/lightning/o/Student_c/new?count=3&nooverride=1&useRecord...`. The browser's address bar and tabs are visible. The main content area displays a 'New Student' form. The form is titled 'New Student' and is part of the 'Students' object. It contains several fields: 'Student ID' (auto-generated), 'First Name', 'Last Name', 'Email', 'Whatsapp Number', 'Enrollment Date', 'Date Of Birth', 'Phone Number', and 'Owner' (Kadapa Charan). The form has a 'Cancel' button, a 'Save & New' button, and a 'Save' button. The background shows a sidebar with 'Students' and 'Recently' sections.

#### iv) Meetings

- Rename Record Name : Meeting ID
- Data type : Auto Number
- Display Format : S-{000000}
- Starting Number : 1
- Deployment status : Deployed
- Allow Search : Access



- Add Notes and Attachment : Selected
- Add to Navigation Tab : Selected

Custom fields for this Object are

- Valid from and valid to as label using date datatype.
- Active as label with checkbox datatype to find whether the meeting is active or not.
- Meeting Url as label with URL datatype to login to the session.

Validation rules:

- Valid from date should be less than Current date.
- Valid To date should be less than valid from.

Snapshot of Meeting Custom Object and its Fields:

The screenshot displays the Salesforce Lightning interface for creating a new meeting. The browser address bar shows the URL: `creative-raccoon-juvjnf-dev-ed.lightning.force.com/lightning/o/Meeting__c/new?count=4&nooverride=1&useRecord...`. The navigation bar includes 'Apps', 'Bookmarks', 'Gmail', 'YouTube', 'Maps', 'Other bookmarks', and 'Reading list'. The main navigation bar shows 'Online Training Ma...' and 'Meetings' selected. The 'New Meeting' form is open, showing the following fields:

- Meeting ID**: A text input field.
- Owner**: A dropdown menu showing 'Kadapa Charan'.
- \*Valid from**: A date input field with a calendar icon.
- \*Valid upto**: A date input field showing '2/4/2022' with a calendar icon.
- \*Meeting Url**: A text input field.
- Active**: A checkbox that is currently unchecked.

At the bottom of the form, there are three buttons: 'Cancel', 'Save & New', and 'Save'.

### **Day 3:**

Topics: Relationship fields and Validation Rules

Milestone/Activities:

#### **Relationship Fields:**

##### **1) Lookup Relationship:**

We can say this as One to many fashion. We can create up to 40 fields in an object. It is not a mandatory, if we delete parent field it does not affect to child field.

##### **2) Master-Detail Relationship:**

In this field, It allows the parent record to control child record attributes such as sharing and visibility. It is always required field. If we delete parent field automatically child field will be deleted.

##### **iv) Batch Scheduling:**

- Rename Record Name : Batch Scheduling ID
- Data type : Auto Number
- Display Format : BE-{000000}
- Starting Number : 1
- Deployment status : Deployed
- Allow Search : Access
- Add Notes and Attachment : Selected
- Add to Navigation Tab : Selected

Custom Fields for this Object are

- Batch ID as label with Autonumber datatype.
- Batch Type as label with Picklist datatype values contains Weekend/Weekday.
- Add lookup fields Course ID to Course, meeting ID to meeting, Trainer ID to Trainer.
- Add two Fields with datatype Date that Start Date and End Date.

##### **v) Course-Trainer Object**

- Rename Record Name : Course-Trainer ID
- Data type : Auto Number
- Display Format : CT-{000000}
- Starting Number : 1
- Deployment status : Deployed
- Allow Search : Access

## Custom Fields for Course Trainer Object

- Add a Master-Detail Relationships for Trainer and Course with label Trainer ID and Course ID.
- Label as Course-Trainer ID by Auto Number datatype.
- Use Formula datatype to display Course Name and Trainer Name.

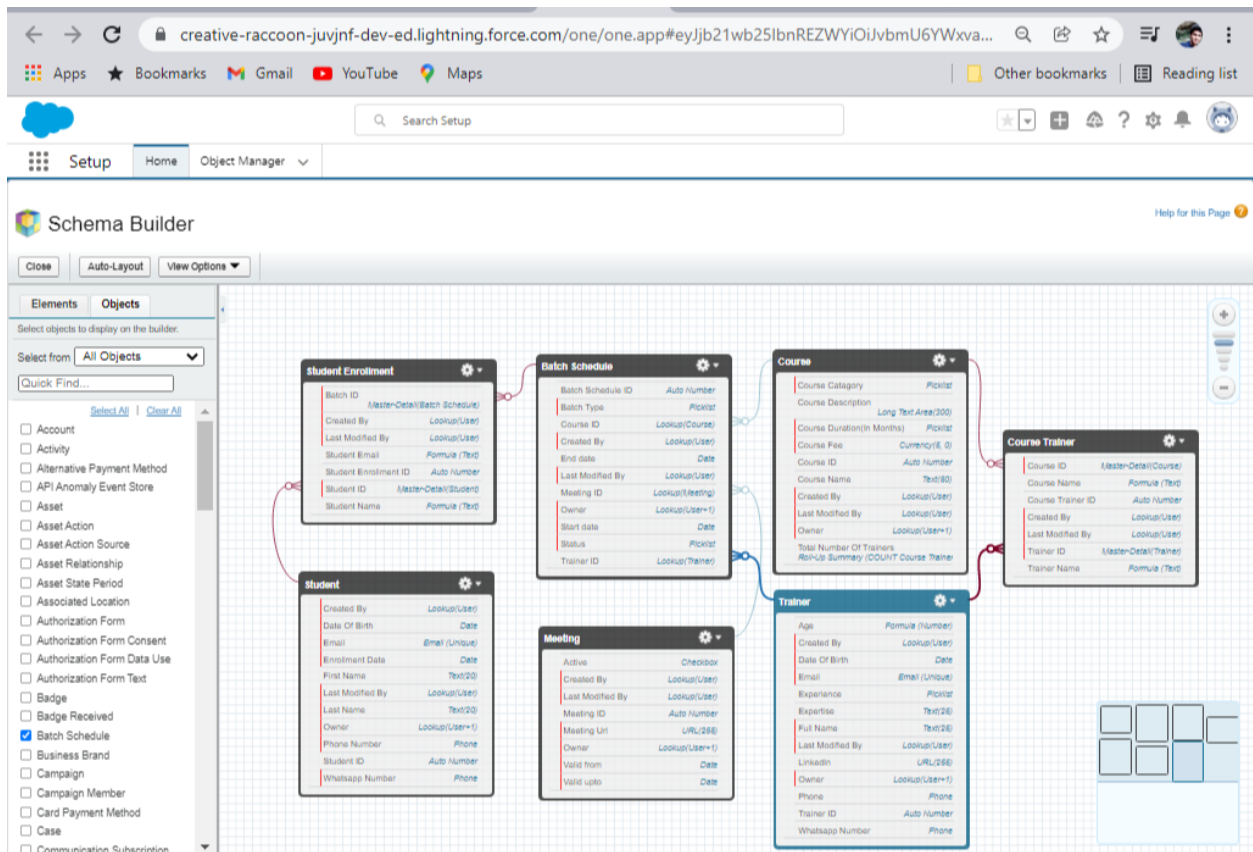
## vi) Student Enrolment Object

- Rename Record Name : Student Enrolment ID
- Data type : Auto Number
- Display Format : SE-{000000}
- Starting Number : 1
- Deployment status : Deployed
- Allow Search : Access

## Custom Fields are

- Add a Master-Detail Relationship with Batch Schedule and Student and label it as Batch ID and Student ID.
- Label as Student Enrolment ID b Auto Number datatype.
- Use formula datatype to display Student Email and Student Name.

## Milestone/Activities screenshot:



## **Day 4:**

Topic : Automations

Milestone/Activities : To reduce the admin efforts salesforce introduces automations which helps to stop doing repetitive tasks. Automations are triggered and executed when the user requirement should be true. This automation works on back end.

Automations are 6 types

The following are GUI based Automation

- 1) Workflow
- 2) Process Builder
- 3) Approval Process
- 4) Flow Automation

The following are Code based Automations

- 5) Trigger
- 6) Apex Schedules

### **1) Workflow**

This works with business logic to automate activities based on some business criteria. When the conditions are true, the automated activity gets executed. Workflow allows you to set various actions based on events or conditions. The actions that can be set include updating a field, creating a task, triggering an email and more. Actions can be triggered based on time of day even.

### **2) Process builder Automation**

Process Builder is an automated Salesforce tool that allows you to control the order of actions or evaluate the criteria for a record. It is a flow Process. Process Builder gives you the ability to use straightforward 'If/Then' logic. In this we can update field, send an email alert, create a record by scheduled time event.

### **3) Approval Process**

Approval Process in salesforce is used for approve records. Records submitted by this user which also called Approver, In Approval Process contains several actions to approve the records, they are

- i) Initial Submission action : It occurs when user first submit the records for approval. It runs automatically. This includes Field update, Email Alert, tasks, and bound messages.
- ii) Final Approval action : It occurs when a record approved from all the approval steps.

This includes Field update, Email Alert, tasks, and bound messages.

iii) Final Rejection action : It occurs when a record rejected by any approval step. This includes Field update, Email Alert, tasks, and bound messages.

iv) Recall action : It occurs when a record recalled after submission for approval. This includes Field update, Email Alert, tasks, and bound messages.

#### 4) Flows

It allows us to build complex business automation by building applications, instead using code. Flows execute logic by call Apex classes. It collects data and perform actions.

Requirement for project:

--> After 1 day of validity of meeting, The Active checkbox automatically be Uncheck using workflow rules.

-->Setting reminder email to meeting owner that 1 day before the valid to date using workflow rule.

--> Sending an email to the batch schedule owner 2 days before the valid to date using workflow rule.

Snapshots for Milestone/Activities:

**Workflow Rules**

Workflow Services

- Batch Management
- Monitor Workflow Services

Process Automation

- Workflow Actions
  - Email Alerts
  - Field Updates
  - Outbound Messages
  - Send Actions
  - Tasks

Workflow Rules

11:41:11 AM

### Edit Rule Active be uncheck after 1day of validto

Help for this Page

Step 3: Specify Workflow Actions Step 3 of 3

Done

Specify the workflow actions that will be triggered when the rule criteria are met. [See an example](#)

Rule Criteria	TODAY()-Valid_upto__c = 1
Evaluation Criteria	Evaluate the rule when a record is created, and any time it's edited to subsequently meet criteria

**Immediate Workflow Actions**

Action	Type	Description
<a href="#">Edit</a>   <a href="#">Remove</a>	Field Update	Active Be Uncheck

Tasks

Workflow Rules

User Interface

Translation Workbench

Export

Import

Override

Translate

Translation Language Settings

Environments

Monitoring

Time-Based Workflow

Security

Expire All Passwords

SETUP

Workflow Rules

Edit Rule Reminder to owner before 1 day

Help for this Page

Step 3: Specify Workflow Actions

Step 3 of 3

Done

Specify the workflow actions that will be triggered when the rule criteria are met. [See an example](#)

Rule Criteria

Valid\_upto\_\_c - TODAY () = 1

Evaluation Criteria

Evaluate the rule when a record is created, and any time it's edited to subsequently meet criteria

Immediate Workflow Actions

Action	Type	Description
<a href="#">Edit</a>   <a href="#">Remove</a>	Email Alert	Email alert to reminder of owner

Add Workflow Action

SETUP

Workflow Rules

Edit Rule Reminder to owner before 2 days of valid upto date

Help for this Page

Step 3: Specify Workflow Actions

Step 3 of 3

Done

Specify the workflow actions that will be triggered when the rule criteria are met. [See an example](#)

Rule Criteria

Valid\_upto\_\_c - TODAY () = 2

Evaluation Criteria

Evaluate the rule when a record is created, and any time it's edited to subsequently meet criteria

Immediate Workflow Actions

Action	Type	Description
<a href="#">Edit</a>   <a href="#">Remove</a>	Email Alert	Email reminder to owner before 2 days of valid to

Add Workflow Action

## Day 5 :

Topic : Custom Page creation using VisualForce

Milestone/activities : Page Creation using VisualForce

### Visual force :

Visual force page is a web pages. These are created by unique tag based Mark-up language. It just like HTML, it's use is to access, display and update the organization's data. This will be accessed by URL link.

Visual-force pages can created by using two ways.

One way--> Go to Developer console--> New--> visual-force page.

Second way--> Setup--> Quick find box--> visual-force pages--> New.

Requirement for Project:

--> Create Course and Trainer pages using Custom Controller.

--> Create Student, Meeting and Batch Schedule pages using Standard Controller.

Snapshots of Milestone/ Activities:

## Course Page

The screenshot shows a web browser window with the URL `creative-raccoon-juvjnf-dev-ed--c.visualforce.com/apex/Course`. The page has a header with navigation links: Apps, Bookmarks, Gmail, YouTube, Maps, Other bookmarks, and Reading list. The main content area is titled "New Course" and contains two sections: "Information" and "Course Description".

**Information Section:**

- Course Name:
- Course Fee \*:
- Course Category \*:
- Course Duration(In Months) \*:

**Course Description Section:**

Course Description

## Trainer Page

The screenshot shows a web browser window with the URL `creative-raccoon-juvjnf-dev-ed--c.visualforce.com/apex/Trainer`. The page has a header with navigation links: Apps, Bookmarks, Gmail, YouTube, Maps, Other bookmarks, and Reading list. The main content area is titled "New Trainer" and contains an "Information" section.

**Information Section:**

- Full Name \*:
- Phone:
- Email \*:
- Date Of Birth \*:  [ 1/8/2022 ]
- Experience:
- Expertise:
- LinkedIn:
- Whatsapp Number:

## Student Page

### New Student

✓ **Success:**  
Record Inserted Successfully

▼ Information

First Name

ramesh

Last Name \*

katta

Date Of Birth \*

12/26/2001

[ 1/8/2022 ]

Email \*

ramesh23@gmail.com

Enrollment Date \*

1/4/2022

[ 1/8/2022 ]

Phone Number \*

(425) 638-9752

Whatsapp Number \*

(425) 638-9752

Save Record

## Meeting Page

← → ↺ creative-raccoon-juvjnf-dev-ed--c.visualforce.com/apex/Meeting

Apps ★ Bookmarks Gmail YouTube Maps | Other bookmarks Reading list

### New Meeting

✓ **Success:**  
Record Inserted Successfully

▼ Information

Valid from \*

1/19/2022

[ 1/8/2022 ]

Valid To \*

2/25/2022

[ 1/8/2022 ]

Meeting Url \*

clay@231gmail.com

Active

☒

Save Record

## Batch Schedule Page

← → ↺ creative-raccoon-juvjnf-dev-ed--c.visualforce.com/apex/Batch\_Schedule

Apps ★ Bookmarks Gmail YouTube Maps | Other bookmarks Reading list

### New Batch

Batch Type \*

--None--

▼

Start date \*

[ 1/8/2022 ]

Course ID

--None--

Weekday

Weekend

Meeting ID

Q

Trainer ID

Q

Status \*

--None--

▼

Save



## Day 6:

Topic : Creating Custom pages using Lightning Web Components

Milestone\ Activities :

**Lightning Web Components(LWC):** Lightning components are used as new Salesforce user interface framework for developing dynamic web applications for desktop and mobile devices. Here we use JavaScript at the client-side and at the server-side is used in Apex to provide the data and business logic to build lightning components. It's an easier way to build an application.

Custom Trainer Page using LWC

for this we need to create an apex class and this apex class is controlled by javascript language using **@AuraEnabled**. Page style can be modified by using CSS.

### Apex Class Code for Trainer Component:

```
public class OnlineTrainingComponentClass {

    @AuraEnabled
    public static String addTrainer(String Trainer_Name,String Trainer_Email,Date
Trainer_DOB,String Trainer_Phone,
                                String Trainer_whatsapp, String Trainer_Expertise,String
Trainer_Experience,String Trainer_LinkedIn)
    {
        Trainer__c t = new Trainer__c();
        t.Full_Name__c = Trainer_Name;
        t.Email__c = Trainer_Email;
        t.Date_Of_Birth__c = Trainer_DOB;
        t.Phone__c = Trainer_Phone;
        t.Whatsapp_Number__c = Trainer_whatsapp;
        t.Expertise__c = Trainer_Expertise;
        t.Experience__c = Trainer_Experience;
        t.Linkedin__c = Trainer_LinkedIn;

        insert t;
        return 'SUCCESS';
    }
}
```

---

**Trainer Component Code:**

```
<aura:component controller="OnlineTrainingComponentClass"
implements="force:appHostable,flexipage:availableForAllPageTypes,forceCommunity:a
vailableForAllPageTypes" access="global" >
```

```
    <aura:attribute name="T_Name" type="String"/>
    <aura:attribute name="T_Date" type="Date"/>
    <aura:attribute name="T_Email" type="String"/>
    <aura:attribute name="T_Phone" type="String"/>
    <aura:attribute name="T_whatsapp" type="String"/>
    <aura:attribute name="T_linkedin" type="String"/>
    <aura:attribute name="T_Expertise" type="String"/>
    <aura:attribute name="T_experience" type="String"/>
    <lightning:notificationsLibrary aura:id="notifLib"/>
```

```
<div class="slds-box">
    <center><h1>Form to Register New Trainer</h1></center>
</div>
```

```
<div class="slds-grid slds-gutters slds-m-top_large ">
    <div class="slds-col slds-size_1-of-2">
    <span><lightning:input name="TName"
        label="Full Name"
        value="{!v.T_Name}"
        placeholder="Enter name here..."/></span>
    </div>
    <div class="slds-col slds-size_1-of-2">
    <span><lightning:input type="date"
        name="TDate"
        value="{!v.T_Date}"
        label="Date of Birth" /></span>
    </div>
</div>
```

```
<div class="slds-grid slds-gutters slds-m-top_large ">
```

```
<div class="slds-col slds-size_1-of-2">
<span><lightning:input type="email"
      name="Temail"
      value="{!v.T_Email}"
      placeholder="Type your Email Here.."
      label="Email Id" /></span>
</div>
```

```
<div class="slds-col slds-size_1-of-2">
<span><lightning:input type="url"
      label="LinkedIn"
      value="{!v.T_linkedin}"
      name="LinkedIn" /></span>
</div>
</div>
```

```
<div class="slds-grid slds-gutters slds-m-top_large ">
  <div class="slds-col slds-size_1-of-2">
    <span><lightning:input type="tel"
          label="Phone"
          value="{!v.T_Phone}"
          name="TPhone"
          pattern="[0-9]{3}-[0-9]{3}-[0-9]{4}" /></span>
    </div>
    <div class="slds-col slds-size_1-of-2">
      <span><lightning:input type="tel"
            label="Whatsapp Number"
            value="{!v.T_whatsapp}"
            name="TPhone"
            pattern="[0-9]{3}-[0-9]{3}-[0-9]{4}" /></span>
    </div>
  </div>
```

```
<div class="slds-grid slds-gutters slds-m-top_large ">
  <div class="slds-col slds-size_1-of-2">
    <span> <lightning:input type="number"
      name="Experience"
      value="{!v.T_Expertise}"
```

```

        label="Experience in Years" /></span>
    </div>
    <div class="slds-col slds-size_1-of-2">
    <span><lightning:input name="Expertise"
        label="Expertise"
        value="{!v.T_experience}"
        placeholder="Select None"/></span>
    </div>
</div>

<center class="slds-m-top_medium" >
    <lightning:button variant="brand"
        label="Save Record"
        title="Save Record"
        onclick="{! c.saveRecord }" />
</center>

</aura:component>

```

---

### **Trainer Component Java Script Code:**

//Save data into appropriate fields in Trainer Object and notify if record created.

```

({
    saveRecord : function(component, event, helper)
    {
        var action=component.get("c.addTrainer");
        action.setParams({
            Trainer_Name:component.get("v.T_Name"),
            Trainer_Email:component.get("v.T_Email"),
            Trainer_DOB:component.get("v.T_Date"),
            Trainer_Phone:component.get("v.T_Phone"),
            Trainer_whatsapp:component.get("v.T_whatsapp"),
            Trainer_Expertise:component.get("v.T_Expertise"),
            Trainer_Experience:component.get("v.T_experience"),
            Trainer_LinkedIn:component.get("v.T_linkedin")
        })
    }
})

```

```

});

action.setCallback(this,function(response){
    var result = response.getReturnValue();
    if(result=='SUCCESS')
    {
        component.find('notifLib').showToast({
            "title": "Success!",
            "message": "The record inserted successfully."
        });
        component.set("v.T_Name","");
        component.set("v.T_Email","");
        component.set("v.T_Date","");
        component.set("v.T_Phone","");
        component.set("v.T_whatsapp","");
        component.set("v.T_Expertise","");
        component.set("v.T_experience","");
        component.set("v.T_linkedin","");
    }
    else
    {
        alert('Record Failed to Insert');
    }
});

$A.enqueueAction(action);
}
})

```

---

### **Trainer Code in CSS:**

//To apply styles to component

```

.THIS {
    background-color:white;
}

```

```
.THIS h1{
  font-size:30px;
  color:red;
  font-weight:bold;
}
```

---

## Snapshot of Milestone/Activities:

Online Training Ma... Home Course New Trainer Student Meeting Batch Schedule

Success! The record inserted successfully.

Full Name  Date of Birth

Email Id  LinkedIn

Phone  Whatsapp Number

Experience in Years  Expertise

## Search in Course Component :

It auto populate fields, name etc., according to search key words.

Apex Code for Search:

@AuraEnabled

```
public static List<Course__c> fetchCourse(String course_str)
{
    String searchstr='%'+ course_str + '%';
    List<Course__c> lstCourse = new List<Course__c>();
    lstCourse=[select Name, Course_Name__c,
Course_Fee__c,Course_Catagory__c,Course_Duration_In_Months__c
    from Course__c where Course_Name__c like : searchstr];
    return lstCourse;
}
```

```
}
```

---

### **Component Controller by Class:**

```
<aura:component controller="OnlineTrainingComponentClass"
implements="force:appHostable,flexipage:availableForAllPageTypes,flexipage:available
ForRecordHome,force:hasRecordId" access="global" >
    <aura:attribute name="cName" type="String"/>
    <aura:attribute name="data" type="Object"/>
    <aura:attribute name="columns" type="List"/>

    <div>
        <lightning:input name="input3"
            label="Enter Course Name"
            value="{!v.cName}"
            onchange="{!c.myAction}"
            placeholder="Enter name here..."/>
    </div>
    <div>
        <lightning:datatable
            keyField="id"
            data="{!v.data}"
            columns="{!v.columns}"
            hideCheckboxColumn="true"/>
    </div>
</aura:component>
```

---

Snapshot of Milestone/Activities:

Welcome to xStudent Dash xSI-7277-1641 xhow to view i xManage Chat xLightning Exp xLightning Co xAura x

creative-raccoon-juvjnf-dev-ed.lightning.force.com/c/SearchCourse.app

Enter Course Name

sa

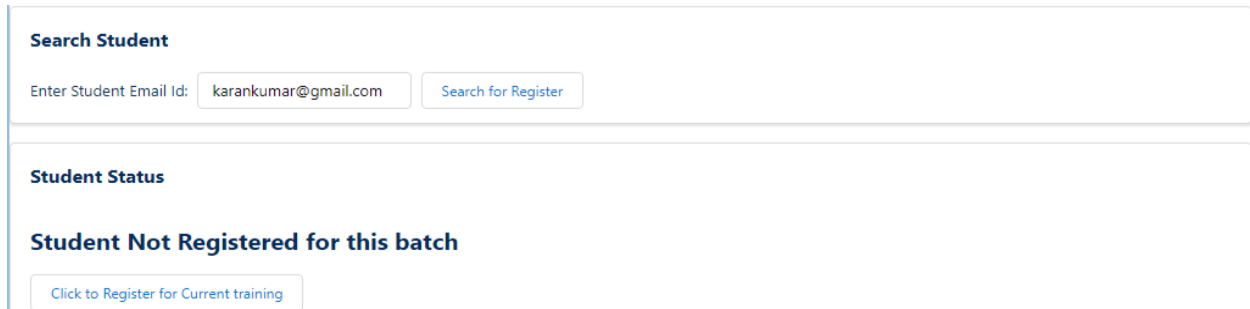
Course Id	Full Name	Course Fee	Course Category	Course Duration(In Months)
a015g00000aYafEAAS	Salesforce Admin	3000	Machine Learning	1
a015g00000aYafIAAC	Salesforce tab	2560	Machine Learning	2
a015g00000aYafJAAS	Salesforce Devop	2400	AWS Cloud Computing	1



## Day 7:

Topic: Search Student and Triggers

Milestone/Activities: Search Student by email and get enrolled if he/she not enrolled before. That will be displayed only students who are not enrolled.



**Search Student**

Enter Student Email Id:  [Search for Register](#)

**Student Status**

**Student Not Registered for this batch**

[Click to Register for Current training](#)

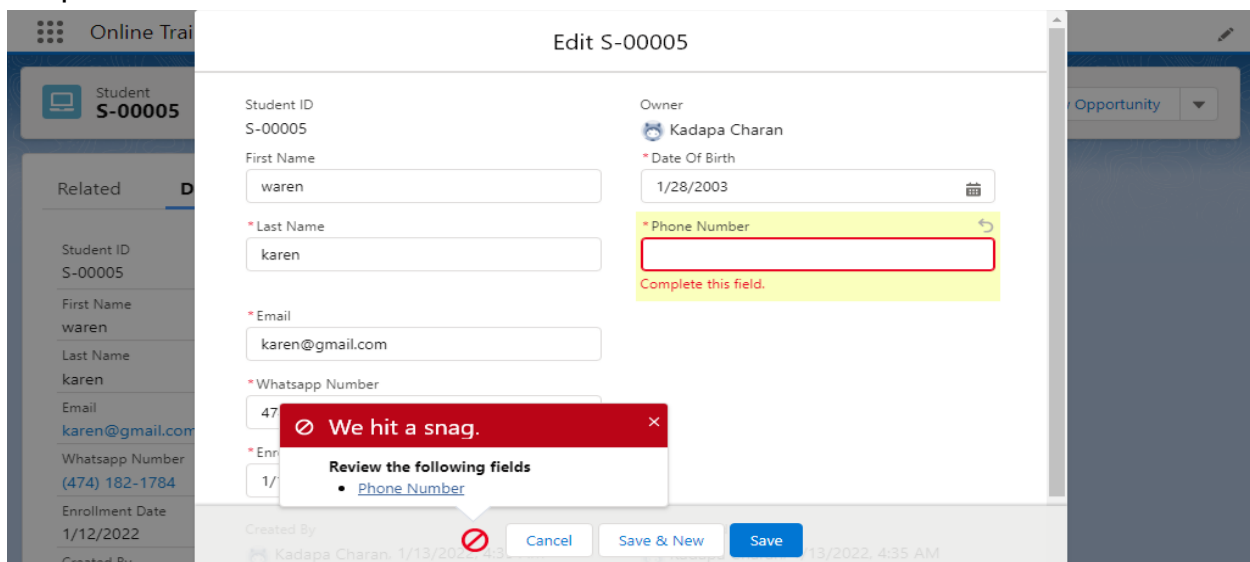
Apex Triggers are enables us to perform an actions at a time which according to record inserted and deleted.

Actions are performed at following events

- before insert
- after insert
- before delete
- after delete
- before update
- after update
- Trigger.New
- Trigger.Old

--> Requirement: Using trigger we modified that phone number and whatsapp number same even student entered different number. And phone number should not be empty.

Snapshot of Milestone/Activities:



**Edit S-00005**

Student ID: S-00005

Owner: Kadapa Charan

\* Date Of Birth: 1/28/2003

\* Phone Number:  Complete this field.

\* Email: karen@gmail.com

\* Whatsapp Number: 47

\* Enr: 1/

**We hit a snag.**


Review the following fields

- [Phone Number](#)








Created By: Kadapa Charan 1/13/2022 4:35 AM


[Cancel](#) [Save & New](#) [Save](#)


Here Whatsapp number is changed which is phone number.





All ▾












Online Training Ma...[Home](#)[Courses ▾](#)[Trainers ▾](#)[Students ▾](#)[Meetings ▾](#)[Batch Schedules ▾](#)

Student  
**S-00005**

 Student "S-00005" was saved.

[Edit](#) [New Opportunity ▾](#)

[Related](#) [Details](#)

Student ID	S-00005	Owner	 <a href="#">Kadapa Charan</a> 
First Name	waren 	Date Of Birth	1/28/2003 
Last Name	karen 	Phone Number	<a href="#">(474) 182-1784</a> 
Email	<a href="mailto:karen@gmail.com">karen@gmail.com</a> 		
Whatsapp Number	<a href="#">(474) 182-1784</a> 		
Enrollment Date	1/12/2022 		

## Day 8:

Topic: Apex Triggers and Student Enrolment by visual pages.

Milestone/Activities:

- 

Requirement for Trainer Object:

--> Trainer should not be able to delete, while the batch is not completed.

### Apex Trigger Code:

```
trigger ValidateDeleteOnTrainer on Trainer__c (before delete) {
    Trainer__c t = Trigger.old[0];
    List<Batch_Schedule__c> lstBatch = [select Name,Status__c from Batch_Schedule__c
                                        where Trainer_ID__c = : t.Id];
    Integer flag=0;
    for(Batch_Schedule__c b : lstBatch)
    {
        if(b.Status__c!='Completed')
        {
            flag=1;
            break;
        }
    }
    if(flag==1)
    {
        t.addError('Cannot delete record, Active batch has found for this trainer');
    }
    else
    {
        List<Messaging.SingleEmailMessage> lstEmail = new
List<Messaging.SingleEmailMessage>();
        Messaging.SingleEmailMessage mail=new Messaging.SingleEmailMessage();
        mail.setToAddresses(new String[] {t.Email__c});
        mail.setReplyTo('kadapacharanru2000@gmail.com');
        mail.setPlainTextBody('Your details removed successfully from our database');
        mail.setSenderDisplayName('Team Charan Project');
        mail.setSubject('Successfully Deleted Trainer');
        lstEmail.add(mail);
        if(lstEmail.size()>0)
```

```

{
  try
  {
    Messaging.sendEmail(IstEmail);
    System.debug('Process Completed Successfully');
  }
  catch(Exception ex)
  {
    System.debug(ex.getMessage());
  }
}

}

```

## Snapshot of Milestone/Activities:

The screenshot shows a Salesforce Lightning page titled "Trainers" with a "Recently Viewed" filter. The page displays a table of 8 trainers. A modal error message is shown over the table, stating: "Cannot delete record, Active batch has found for this trainer".

	Trainer ID	Full Name
1	T-00001	Kamal Hasan
2	T-00009	aadhish
3	T-00008	
4	T-00007	
5	T-00006	redish
6	T-00005	ramseh
7	T-00004	ramus
8	T-00003	shanool

## Day 9:

Topic: Student Enrolling to the Scheduled batches through by student email.

Milestone/Activities :

Arranging table that batches scheduled, and enrolling students to those batches if they are not. If already enrolled, alerting students by notifying warning message.

TrainingCalendar VF page which controlled by TrainingCalendarApex page to show the scheduled batches in a table format. And also adding a Command Button to **Register** to those Scheduled batches.

TrainingRegistration VF page controlled by TrainerRegistrationApex page performs further steps after selecting a scheduled batch by which course you want to register, then it should redirects to another page by showing the course name and batch id. Here we should first check whether the student already enrolled to this batch or not by entering student email. If the student are new, then it will notify the Not registered, so we can enroll by click below Command button. If email already enrolled then it will notifies the student by warning message.

### TrainingCalendar Code:

```
<apex:page controller="TrainingCalendarApex" lightningStylesheets="true">
<apex:form >
    <apex:sectionHeader title="Training List"/>

    <apex:pageBlock id="bob">
        <apex:pageBlockTable value="{!lstWrap}" var="t">
            <apex:column value="{!t.key}" headerValue="Key"/>
            <apex:column value="{!t.batchInfo.Name}" headerValue="Batch Number"/>
            <apex:column value="{!t.batchInfo.Course_Name__c}" headerValue="Course
Name"/>
            <apex:column value="{!t.batchInfo.Batch_Type__c}" headerValue="Batch Type"/>
            <apex:column value="{!t.batchInfo.Status__c}" headerValue="Batch Status"/>
            <apex:column headerValue="Action">

                <apex:commandButton value="Register" action="{!Register}" reRender="bob">
                    <apex:param name="KeyToRegister" value="{!t.key}"
assignTo="{!keyToRegister}"/>
                </apex:commandButton>
            </apex:column>
        </apex:pageBlockTable>
    </apex:pageBlock>
</apex:form>
</apex:page>
```

```

        </apex:commandButton>

    </apex:column>
</apex:pageBlockTable>

</apex:pageBlock>

</apex:form>
</apex:page>

```

---

### **TrainingCalendarApex :**

```

public with sharing class TrainingCalendarApex
{

```

```

    public List<wrapper> lstWrap{get;set;}
    List<Batch_Schedule__c> lstBatch;
    public Integer num;

```

```

    public Integer keyToRegister{get;set;}

```

```

    public TrainingCalendarApex()
    {
        num=1;
        keyToRegister=0;
        lstWrap=new List<wrapper>();
        lstBatch=[select Name,Course_Name__c,Batch_Type__c,Status__c
                    from Batch_Schedule__c where Status__c =: 'Scheduled'];
        for(Integer i=0;i<lstBatch.size();i++)
        {
            lstWrap.add(new wrapper(num++,lstBatch[i]));
        }
    }

```

```
}
```

```
public PageReference Register()
{
    Boolean found=false;
    Id recordId=null;

    for(wrapper w:lstWrap)
    {
        if(w.key == keyToRegister)
        {
            found=true;
            recordId=w.batchInfo.Id;
            break;
        }
    }
    if(found)
    {
        PageReference pr=new
PageReference('/apex/TrainingRegistration?id='+recordId);
        return pr;
    }
    return null;
}
```

```
public class wrapper
{
    public Integer key{get;set;}
    public Batch_Schedule__c batchInfo{get;set;}

    public wrapper(Integer key,Batch_Schedule__c batchInfo)
    {
        this.key=key;
```

```

        this.batchInfo=batchInfo;
    }

}
}

```

---

### TrainingRegistration Code:

```

<apex:page controller="TrainerRegistrationApex" lightningStyleSheets="true">
  <apex:form >
    <apex:sectionHeader title="Register for Training"/>
    <apex:pageMessages ></apex:pageMessages>

    <h1 style="font-size:15px;">Register for:#{!batchInfo[0].Name}</h1> <p/>
    <h1 style="font-size:15px;">Course Name:#{!batchInfo[0].Course_Name__c}</h1>

    <apex:pageBlock title="Search Student">
      Enter Student Email Id: &nbsp;<apex:inputText value="{!studEmail}"/>
      <apex:commandButton value="Search for Register" action="{!searchStudent}"/>

    </apex:pageBlock>

    <apex:pageBlock rendered="{!studexist}" >
      <apex:commandButton value="Click to Register" action="{!registerStudent}"/>

    </apex:pageBlock>

    <apex:pageBlock title="Student Status" rendered="{!display1}">
      <h1 style="font-size:20px;">Student Not Registered for this batch</h1>
      <apex:commandButton value="Click to Register for Current training"
action="{!clicktoRegister}"/>

    </apex:pageBlock>

```



```
</apex:form>
</apex:page>
```

---

### **TrainerRegistrationApex:**

```
public with sharing class TrainerRegistrationApex {

    public Id recId{get;set;}
    public List<Batch_Schedule__c> batchInfo{get;set;}
    public String studEmail{get;set;}
    public List<Student__c> studList{get;set;}
    public boolean studexist{get;set;}
    public boolean display1{get;set;}
    public List<Student_Enrollment__c> studEnroll;

    //Construtor
    public TrainerRegistrationApex ()
    {

        recId=ApexPages.currentPage().getParameters().get('id');
        batchInfo=[SELECT  Status__c,Batch_Type__c,Course_Catagory__c,
                        Course_ID__c,Course_Name__c,End_date__c,Meeting_ID__c,
                        Meeting_URL__c,Name,Start_date__c,Trainer_Email__c,Trainer_ID__c,
                        Trainer_Name__c FROM Batch_Schedule__c where Id=: recId];
        studexist=false;
        display1=false;
    }

    public PageReference searchStudent()
    {
        Integer found=0;

        if(studEmail==" || studEmail==null)
        {
            ApexPages.addMessage(new ApexPages.message(ApexPages.severity.Error,'No
```

```

Email Id Provided for Search'));
    studexist=false;
}
else
{
    studList=[SELECT Date_Of_Birth__c,Email__c,
        Enrollment_Date__c,First_Name__c,Id,
        Last_Name__c,Name,Phone_Number__c,Whatsapp_Number__c FROM
Student__c where Email__c =: studEmail];

    if(studList.size()>0)
    {
        studEnroll=[SELECT Batch_ID__c,Id,Name,Student_Email__c,
            Student_ID__c,Student_Name__c FROM Student_Enrollment__c where
Student_ID__c=: studList[0].id];

        if(studEnroll.size()>0)
        {
            for(Student_Enrollment__c s:studEnroll)
            {
                if(s.Batch_ID__c == batchInfo[0].Id)
                {
                    ApexPages.addMessage(new
ApexPages.message(ApexPages.severity.Warning,'Student Already Registered in this
batch'));
                }
            }
        }
        else
        {
            display1=true;
        }
    }
}
else

```

```

        {
            ApexPages.addMessage(new
ApexPages.message(ApexPages.severity.Info,'Student Not Registered in any batch'));
            display1=true;
        }

        studexist=false;
    }
    else
    {
        ApexPages.addMessage(new
ApexPages.message(ApexPages.severity.Error,'No Student Registered with this email
Id'));
        display1=false;
        studexist=true;
    }
}

```

```

return null;
}

```

```

//Function to register student in current batch
public PageReference clicktoRegister()
{
    Student_Enrollment__c s=new Student_Enrollment__c();
    s.Student_ID__c =studList[0].id;
    s.Batch_ID__c= batchInfo[0].id;
    insert s;
    ApexPages.addMessage(new
ApexPages.message(ApexPages.severity.Confirm,'Student Enrollment Done
Successfully'));
    display1=false;
    studexist=false;
}

```

//Custom code to send a acknowledgement email to student

```

        List<Messaging.SingleEmailMessage> lstEmail=new
List<Messaging.SingleEmailMessage>();
        Messaging.SingleEmailMessage mail=new Messaging.SingleEmailMessage();
        mail.setToAddresses(new String[] {studList[0].Email__c});
        mail.setReplyTo('kadapacharanru2000@gmail.com');
        mail.setPlainTextBody('Registered Successfully for Batch
Number'+batchInfo[0].Name);
        mail.setSenderDisplayName('Team Charan Project');
        mail.setSubject('Registered Successfully');
        lstEmail.add(mail);
        if(lstEmail.size()>0)
        {
            try
            {
                Messaging.sendEmail(lstEmail);
                System.debug('Process Completed Successfully');
            }
            catch(Exception ex)
            {
                System.debug(ex.getMessage());
            }
        }

        return null;
    }

```

```

    public PageReference registerStudent()
    {
        return null;
    }
}

```

---

## Snapshots of Milestone/Activities :

### Table of Scheduled Batches

KEY	BATCH NUMBER	COURSE NAME	BATCH TYPE	BATCH STATUS	ACTION
1	B-0002	Salesforce Devop	Weekend	Scheduled	<a href="#">Register</a>
2	B-0003	Salesforce Admin	Weekend	Scheduled	<a href="#">Register</a>

If Student already enrolled, a warning message will pop up

**Warning:**  
Student Already Registered in this batch

Course Name:-Salesforce Devop

**Search Student**

Enter Student Email Id:  [Search for Register](#)

If Student not enrolled, then will display like this

The screenshot shows a web browser window with the URL `creative-raccoon-juvjnf-dev-ed--c.visualforce.com/apex/TrainingCalendar`. The page title is "Register for Training". A dark grey message bar at the top states "Student Not Registered in any batch". Below this, the "Course Name:-Salesforce Devop" is displayed. The "Search Student" section contains an input field with the email "karankumar@gmail.com" and a "Search for Register" button. The "Student Status" section displays the message "Student Not Registered for this batch" and a button labeled "Click to Register for Current training". The browser's taskbar at the bottom shows two open tabs: "TrainingRegistration" and "TrainingCalendar".

And after a success message will pop up

This screenshot shows the same "Register for Training" page after a successful registration. A green message bar at the top displays "Success: Student Enrollment Done Successfully". The "Course Name:-Salesforce Devop" remains the same. In the "Search Student" section, the input field now contains the email "ramesh23@gmail.com". The "Student Status" section is empty. The browser's taskbar at the bottom shows the same two tabs: "TrainingRegistration" and "TrainingCalendar".

## Day 10:

Topic : Reports & Dashboards

Milestone/Activities :

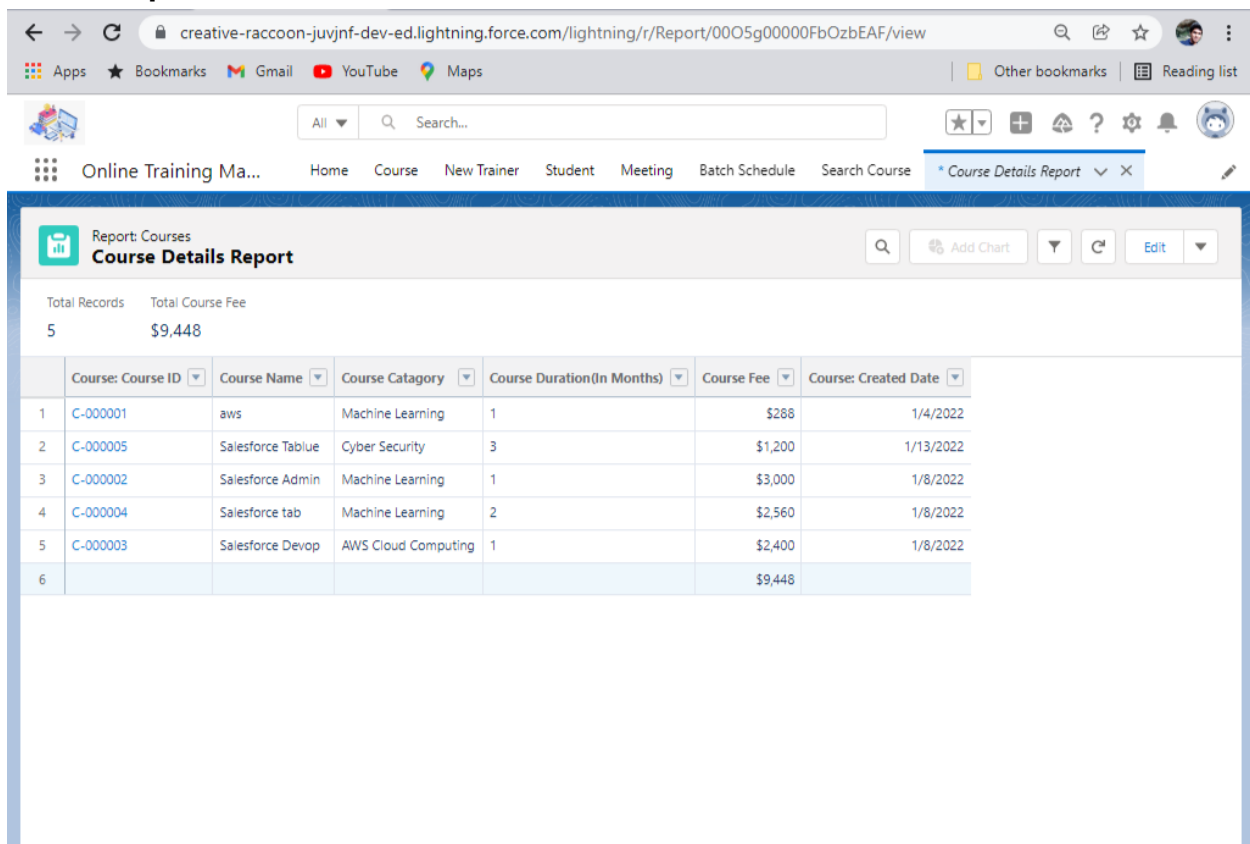
**Reports** : Reports in salesforce used to organize records data in the form of list that meets the user requirement criteria. It is displayed in the form of Tables. We can also show this bar charts etc.

Reports have different types

- 1) Tabula Reports : It is for list all the courses.
- 2) Summary Reports : It is for all Courses are subtotaled by batch type
- 3) Matrix Reports : It is for summarize the batches by batch type.
- 4) Joined Reports : It is for combination of two fields in a single reports which only having master-detail relationship.

Snapshots of Milestone/activities :

### Tabula Report



The screenshot displays a Salesforce Lightning interface for a report titled "Course Details Report". The report is a tabular format showing 5 records. The columns are: Course ID, Course Name, Course Category, Course Duration (in Months), Course Fee, and Course Created Date. The total number of records is 5, and the total course fee is \$9,448.

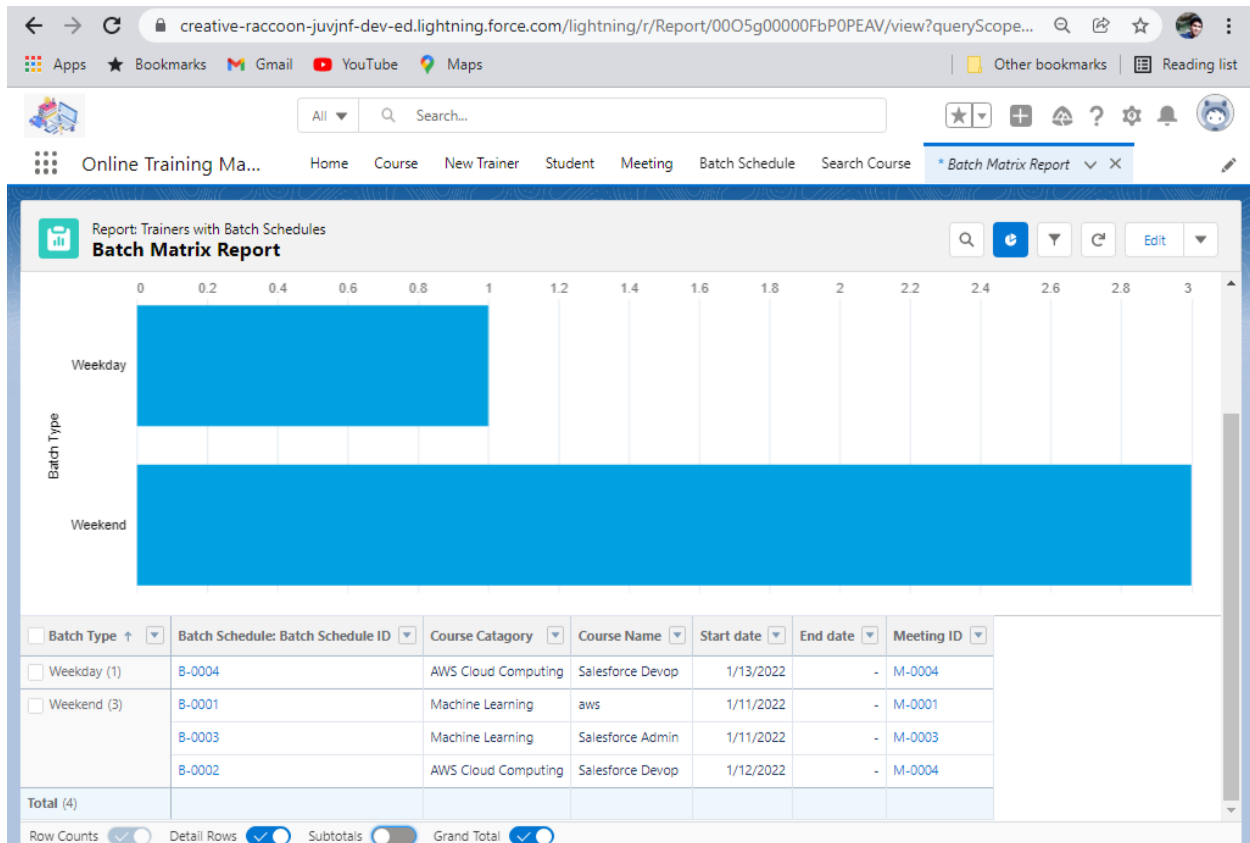
	Course: Course ID	Course Name	Course Category	Course Duration(In Months)	Course Fee	Course: Created Date
1	C-000001	aws	Machine Learning	1	\$288	1/4/2022
2	C-000005	Salesforce Tablue	Cyber Security	3	\$1,200	1/13/2022
3	C-000002	Salesforce Admin	Machine Learning	1	\$3,000	1/8/2022
4	C-000004	Salesforce tab	Machine Learning	2	\$2,560	1/8/2022
5	C-000003	Salesforce Devop	AWS Cloud Computing	1	\$2,400	1/8/2022
6					\$9,448	

## Summary Report

<input type="checkbox"/> Batch Type ↑	Batch Schedule: Batch Schedule ID	Course Category	Course Name	Start date	End date	Meeting ID
<input type="checkbox"/> Weekday (1)	B-0004	AWS Cloud Computing	Salesforce Devop	1/13/2022	-	M-0004
Subtotal						
<input type="checkbox"/> Weekend (3)	B-0001	Machine Learning	aws	1/11/2022	-	M-0001
	B-0003	Machine Learning	Salesforce Admin	1/11/2022	-	M-0003
	B-0002	AWS Cloud Computing	Salesforce Devop	1/12/2022	-	M-0004
Subtotal						
Total (4)						

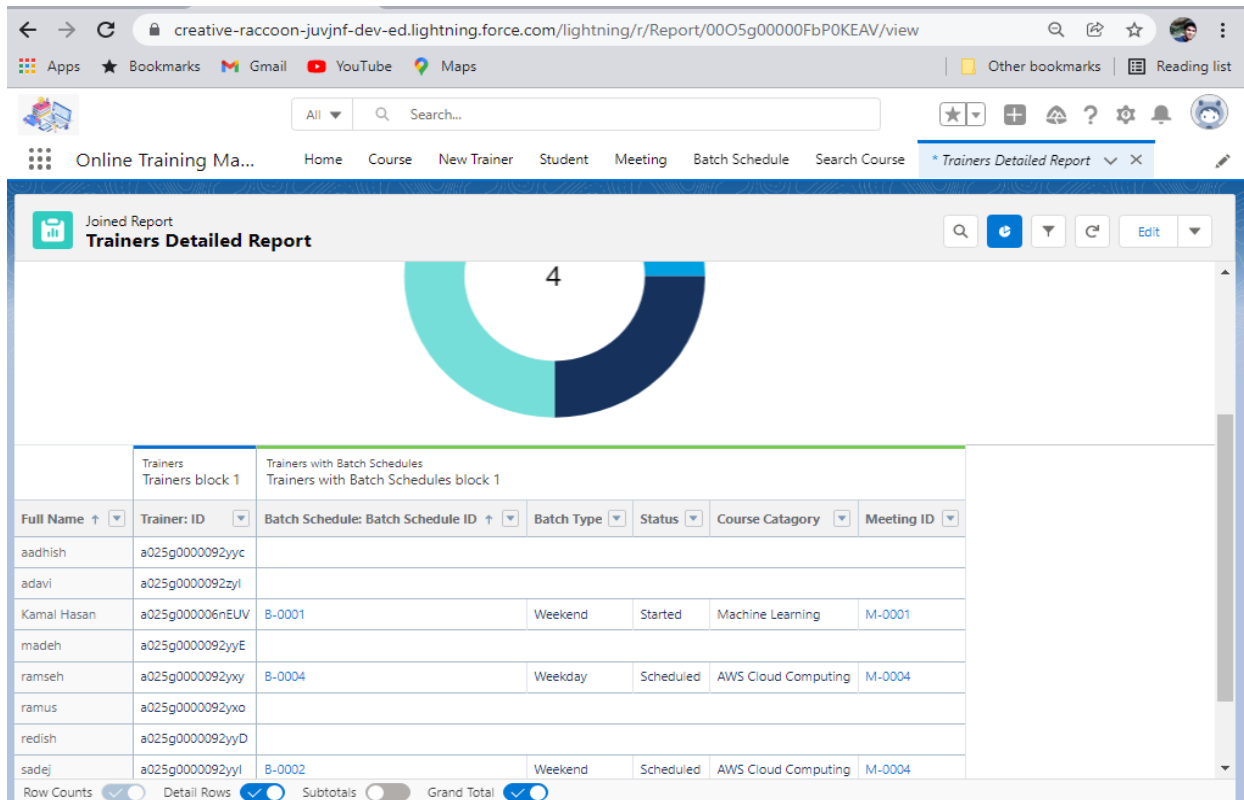
Row Counts ☒ Detail Rows ☒ Subtotals ☒ Grand Total ☒

## Matrix Report





## Joined Report



**Dashboards:** This is used for graphical and visual representation of Reports in salesforce. It have some visual representation components like graphs, charts, gauges, tables, metrics and visualforce pages. Salesforce dashboard components are used to represent data. We can use up to 20 components in single dashboard. Created Dashboards or reports can be stored in folders which we created.

Snapshot of Milestone/Activities :

Dashboard added to Online Training Management Application Home page

