

College Management Application

Day-1:

Create Your Salesforce Developer Org To Get Started

Creating a developer org in salesforce.

Go to [developers.salesforce.com/](https://developer.salesforce.com/)

Click on sign up.

On the sign up form, enter the following details :

First name & Last name

Email

Role : Developer

Company : College Name

Country : India

Postal Code : pin code

Username : should be a combination of your name and company

This need not be an actual email id, you can give anything in the format :

username@organization.com

Click on sign up after filling these.

Account Activation

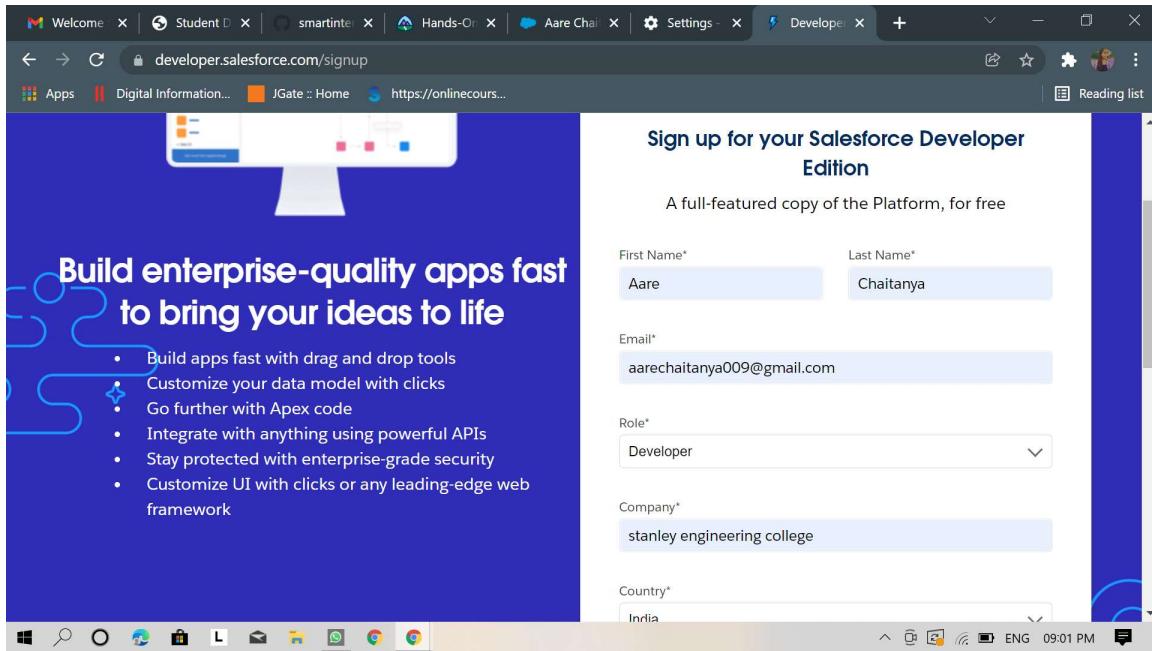
Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins, as allocating a developer org will take time.

Login To Your Salesforce Account

1.Go to [salesforce.com](https://www.salesforce.com) and click on login.

2.Enter the username and password that you just created.

3.After login this is the home page which you will see.



Day 2:

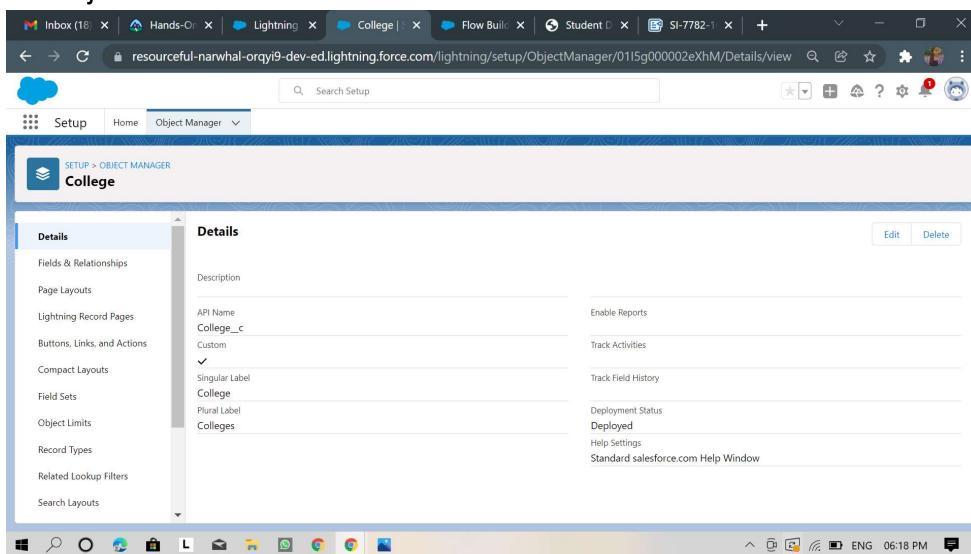
1. (i)Custom Object Creation:

Custom Objects: Custom objects are the objects which are created manually by the users.

Create The Custom Objects For The Application

Create the following Custom Objects For the Application.

1. College
2. Application Form.
3. Students.
4. Subjects.



The screenshot shows the Salesforce Object Manager interface for the 'Student' object. The left sidebar lists various configuration tabs: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, and Search Layouts. The main 'Details' tab displays the following information:

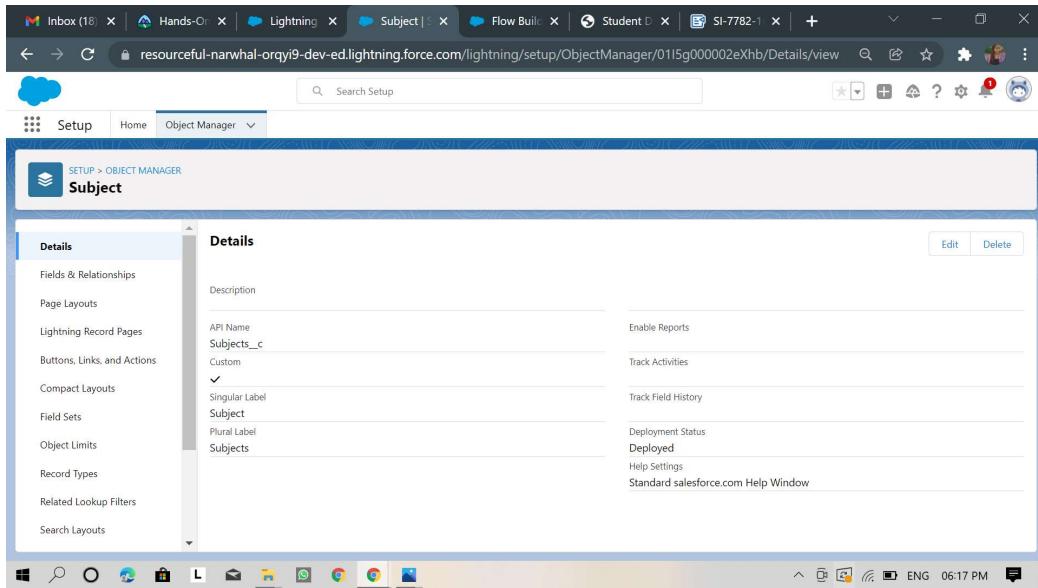
API Name	Description
Student__c	Enable Reports
Custom	Track Activities
Singular Label	Track Field History
Student	Deployment Status
Plural Label	Deployed
Student	Help Settings
	Standard salesforce.com Help Window

At the bottom right of the main area are 'Edit' and 'Delete' buttons.

The screenshot shows the Salesforce Object Manager interface for the 'Application Form' object. The left sidebar lists the same configuration tabs as the Student object. The main 'Details' tab displays the following information:

API Name	Description
Application_Form__c	Enable Reports
Custom	Track Activities
Singular Label	Track Field History
Application Form	Deployment Status
Plural Label	Deployed
Application Forms	Help Settings
	Standard salesforce.com Help Window

At the bottom right of the main area are 'Edit' and 'Delete' buttons.



(ii) Create Fields On College Object

Create the following fields on the college object.

Field Name

Data Type

Required

Values

Record Info

Text

College Fees

Currency(7,2)

Yes

Hostel Fees

Currency(6,2)

Yes

College Name

Picklist(Refer Business Logic In Milestones)

Yes

Email

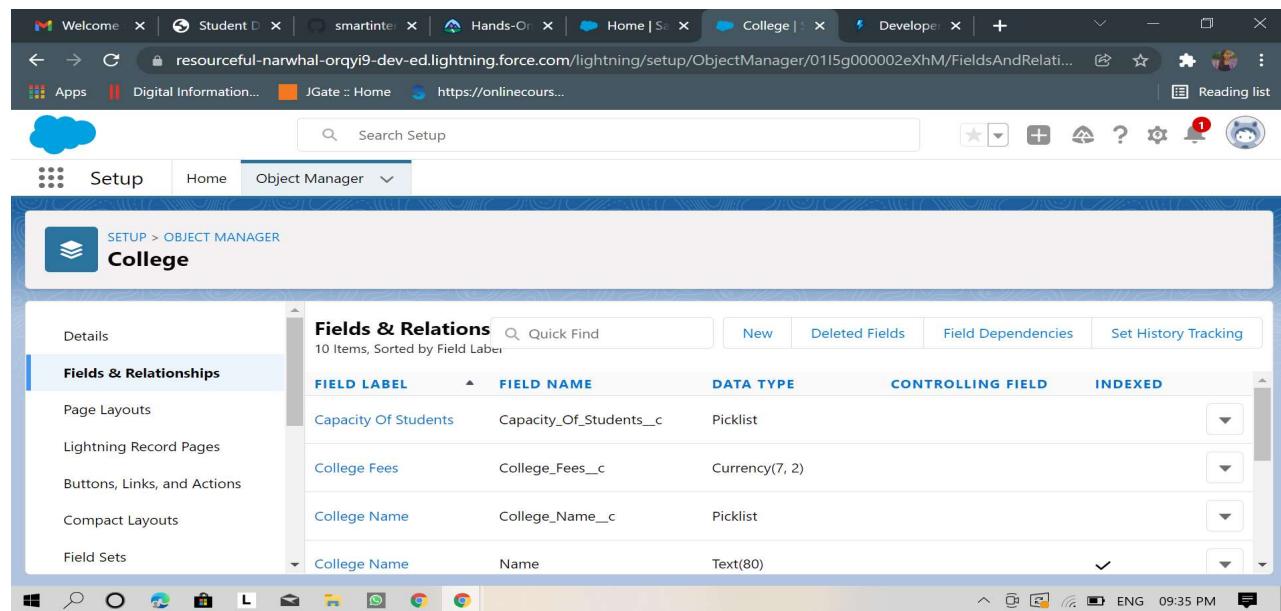
Picklist

b1r@mit.co.in
hyd@mit.co.in
mum@mit.co.in
maa@mit.co.in
ccu@mit.co.in
del@mit.co.in

Capacity Of Students

Picklist

500-1000, 1000-2500, 2500-6000, 6000-10000



The screenshot shows the Salesforce Object Manager interface for the 'College' object. The left sidebar lists various setup categories like Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, and Field Sets. The main area is titled 'Fields & Relations' and displays a table of fields. The table has columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The data in the table is as follows:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Capacity Of Students	Capacity_of_Students__c	Picklist		
College Fees	College_Fees__c	Currency(7, 2)		
College Name	College_Name__c	Picklist		
College Name	Name	Text(80)		

(iii) Create Fields On Application Form Object

Create the following fields on the application form object.

Field Name

Data Type

Required

Values

Application Form ID

Auto-number

F-{000000}

Starting Number=1

Address

Text(255)

Yes

College

Master-Detail(College)

Yes

College Fees

Formula(Currency)

Hostel Fees

Formula(Currency)

date of Birth

Date

Yes

Email

Email(Unique)

Yes

Guardian Name

Text(30)

Yes

Looking For Hostel Stay

Checkbox(default=Uncheck)

Ready To Join

Checkbox(default=Uncheck)

Student Name

Text(30)

Yes

Phone

Phone

Yes

The screenshot shows the Salesforce Object Manager Fields & Relations page for the Application Form object. The page title is "Application Form". On the left, there's a sidebar with links like Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, and Field Sets. The main area is titled "Fields & Relations" and shows a table with the following data:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address_c	Text(255)		
Application Form ID	Application_Form_ID_c	Auto Number		
Application Form Name	Name	Text(80)		✓
College	College_c	Master-Detail(College)		✓

(iv) Create Fields On Student Object

Create the following fields on the student object.

Field Name

Data Type

Required

Values

Student Name

Text

Address

Text(255)

Yes

Application Form

Lookup(Application Form)

College Name

Formula(Text)

Date Of Birth

Date

Yes

Guardian Name

Text(30)

Yes

Phone

Phone

Yes

The screenshot shows the Salesforce Object Manager interface for the 'Student' object. The left sidebar has 'Fields & Relationships' selected. The main area displays a table titled 'Fields & Relations' with the following data:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address_c	Text(255)		
Application Form	Application_Form__c	Lookup(Application Form)		✓
College Name	College_Name__c	Formula (Text)		
Created By	CreatedBy	Lookup(User)		

(v) Create Fields On Subject Object

Create the following fields On the Subject object.

Field Name

Data Type

Required

Values

Subject ID

AutoNumber

S-{000000}

Starting Number=1

Paper 1

Picklist(Refer Business Logic Milestone)

Paper2

Picklist(Refer Business logic Milestone)

Student Lookup(Student)

The screenshot shows the Salesforce Setup interface for the 'Subject' object. The left sidebar has 'Fields & Relationships' selected. The main area displays a table titled 'Fields & Relations' with the following data:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Paper 1	Paper_1__c	Picklist		

2.Adding Business Logic To Application

(i)Creating Global Picklist Value Sets

1. Create the following global picklist value sets for the application.
 - a)College

Picklist Value Name

Values

College

MIT-HYD

MIT-BLR

MIT-MUM

MIT-MAA

MIT-DEL

MIT-CCU

- b)Paper1

Picklist Value Name

Values

Paper 1

APEX

JAVA

C

C++

c)Paper2

Picklist Value Name

Values

Paper2

MATHEMATICS

ENGLISH

STATISTICS

The screenshot shows the Salesforce Lightning interface with the URL <https://resourceful-narwhal-orqy19-dev-ed.lightning.force.com/lightning/setup/Picklists/page?address=%2Fui%2Fplatform%2Fui%2Fpicklist%2FvalueSets>. The page title is "Picklist Value Sets". On the left, there's a sidebar with "Data" and "Objects and Fields" sections, and a search bar. The main content area displays a table titled "Global Value Sets" with three entries:

Action	Label	Description
Edit Del	College	
Edit Del	Paper1	
Edit Del	Paper2	

(ii)Creating Field Dependencies

1. Create field dependency between college Name and Email, where the controlling field is college Name and dependent field is Email. Select the email ids according to the college names.
2. Create field dependency between college Name and capacity of students, where the controlling field is college Name and dependent field is Capacity of Students. Select the values according to your wish.

The screenshot shows the Salesforce Lightning Object Manager interface for the 'College' object. The page title is 'College Field Dependencies'. It displays a table of field dependencies:

Action	Controlling Field	Dependent Field	Modified By
Edit Del	College Name	Email	Aare_Chaitanya, 1/4/2022, 8:57 AM

The sidebar on the left lists various setup options like Details, Fields & Relationships, Page Layouts, and Lightning Record Pages.

The screenshot shows the Salesforce Lightning Object Manager interface for the 'College' object. The page title is 'College'. A dependent picklist for 'Email' is displayed, showing values corresponding to different college names:

College Name:	MIT-BLR	MIT-HYD	MIT-MUM	MIT-MAA	MIT-CCU
Email:	blr@mit.co.in	blr@mit.co.in	blr@mit.co.in	blr@mit.co.in	blr@mit.co.in
	hyd@mit.co.in	hyd@mit.co.in	hyd@mit.co.in	hyd@mit.co.in	hyd@mit.co.in
	mum@mit.co.in	mum@mit.co.in	mum@mit.co.in	mum@mit.co.in	mum@mit.co.in
	maa@mit.co.in	maa@mit.co.in	maa@mit.co.in	maa@mit.co.in	maa@mit.co.in
	ccu@mit.co.in	ccu@mit.co.in	ccu@mit.co.in	ccu@mit.co.in	ccu@mit.co.in
	del@mit.co.in	del@mit.co.in	del@mit.co.in	del@mit.co.in	del@mit.co.in

The sidebar on the left lists various setup options like Details, Fields & Relationships, Page Layouts, and Lightning Record Pages.

The screenshot shows the Salesforce Lightning setup interface. The URL in the address bar is <https://resourceful-narwhal-orqyiq9-dev-ed.lightning.force.com/lightning/setup/ObjectManager/01I5g000002eXhM/FieldsAndRelati...>. The page title is "College Field Dependencies". The left sidebar shows navigation options like "Details", "Fields & Relationships", "Page Layouts", "Lightning Record Pages", "Buttons, Links, and Actions", "Compact Layouts", and "Field Sets". The main content area displays a table titled "Field Dependencies" with two rows:

Action	Controlling Field	Dependent Field	Modified By
Edit Del	College Name	Email	Aare Chaitanya, 1/4/2022, 8:57 AM
Edit Del	College Name	Capacity Of Students	Aare Chaitanya, 1/4/2022, 9:00 AM

(iii) Create Validation Rule On College Object

1. Create a validation rule such that the college name and record info should have the same name.

TEXT(College_Name__c) <> Name

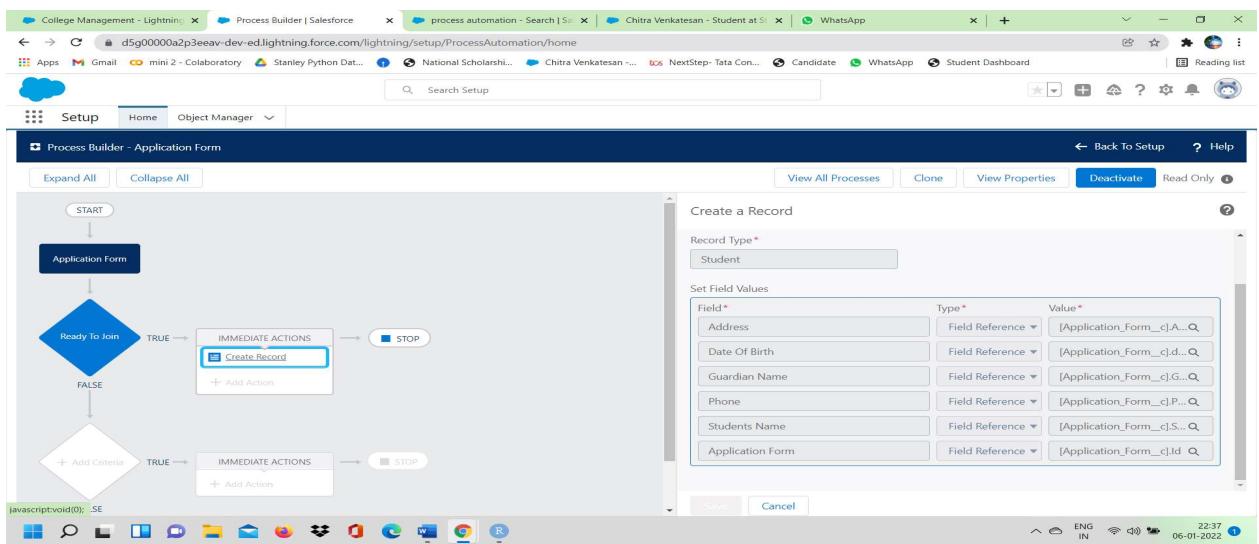
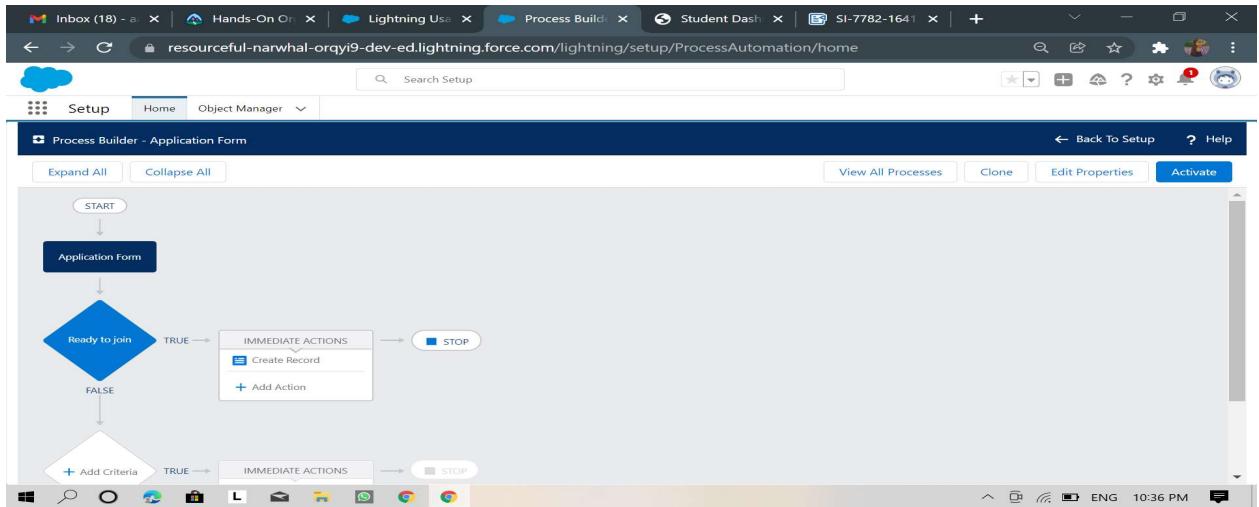
2.. Create a validation rule on the application form object to stop any modification on the application form once a student record is created.

```
AND(Ready_To_Join__==true,
OR(
ISCHANGED(Address__c),
ISCHANGED(College__c),
ISCHANGED(Date_Of_Birth__c),
ISCHANGED(Email__c),
ISCHANGED(Guardian_Name__c),
ISCHANGED(Phone__c)
)
)
```

Day 4:

(iv) Process Automation

1. Create an automation process such that when the "ready to join" field is checked on the application form object we need to create the student record automatically with the information specified in the application form record.
2. Go to Setup -> select "Process Builder" from quick find. Create a Process Builder on the "Application Form" object with a condition as "When a record change". And select "When a record is created or edited".
 - a. In the diamond shape box(called nodes), select the criteria which trigger the Process builder to fire. In our example, it is "When Ready to Join field is checked."
 - b. Once the node is setup, click on the adjacent box called "Immediate action". And select create a record on the student object. Please follow the below screenshot.



Day 5:

Create The Student Record Using Flow:

1. First deactivate the process builder which we created earlier.
2. Now search for flows and select new flow ->record triggered flow
3. For object select application form, in configure trigger select when the record is updated for entry criteria select as ready to join equals to true.
4. Now create a variable named student in the resource section.
5. Now add the assignment as follows.
6. Now add create records.

Configure Start

Set Entry Conditions

Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.

If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the **Only when a record is updated to meet the condition requirements** option for When to Run the Flow for Updated Records.

Condition Requirements

All Conditions Are Met (AND)

Field: Ready_To_Join_c Operator: Equals Value: {!\$GlobalConstant.True}

+ Add Condition

When to Run the Flow for Updated Records

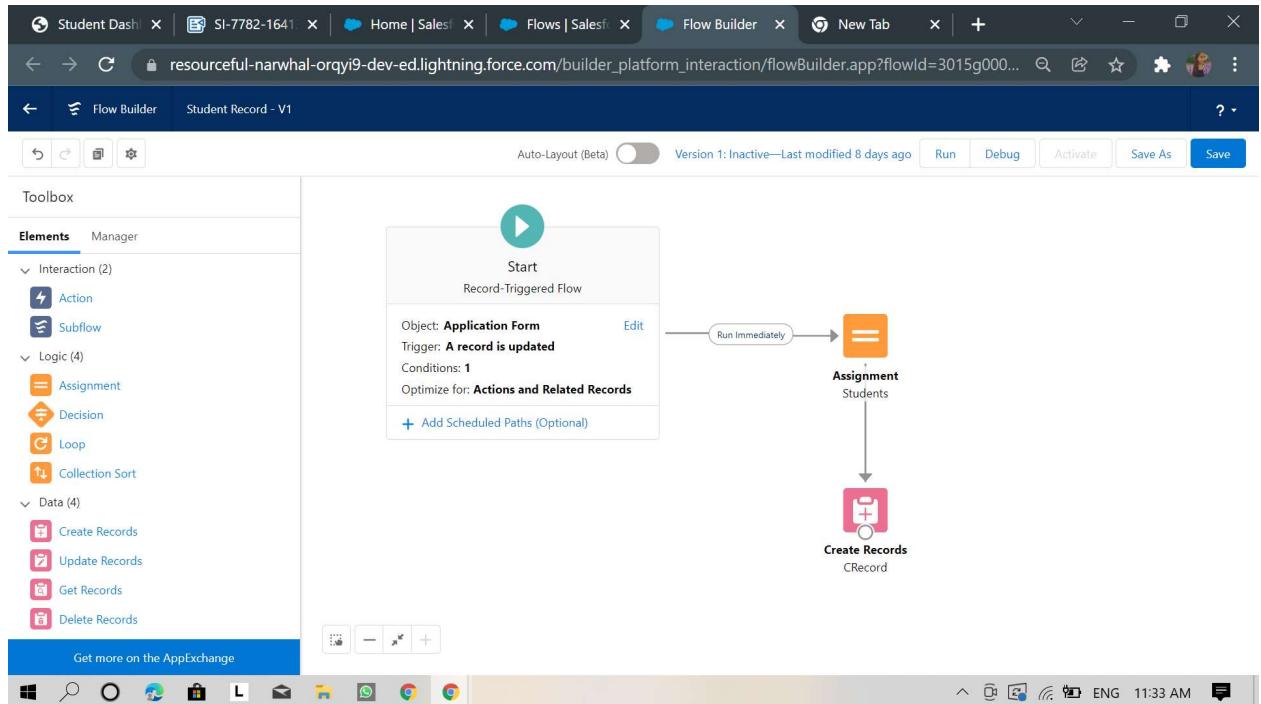
- Every time a record is updated and meets the condition requirements
- Only when a record is updated to meet the condition requirements

Cancel Done

Edit Assignment

Variable	Operator	Value
A ₃ Student > Address	Equals	A ₃ \$Record > Address
A ₃ Student > Application Form	Equals	A ₃ \$Record > Record ID
A ₃ Student > College Name	Equals	A ₃ \$Record > College
A ₃ Student > Date Of Birth	Equals	A ₃ \$Record > date of Birth
A ₃ Student > Guardian Name	Equals	A ₃ \$Record > Guardian Name
A ₃ Student > Phone	Equals	A ₃ \$Record > Phone
A ₃ Student > Student Name	Equals	A ₃ \$Record > Student Name

Cancel Done



Day 6:

- From the developer console create a new apex class and enter the following code.

```
Public class ApplicationBatchTest implements Database.Batchable<sObject>{
```

```
//start(), execute(). finish()
public Integer totalForms = 0; // total no of application form
public Integer totalConvertedForms = 0; // total no of students
public Database.QueryLocator start(Database.BatchableContext bc){
    // gathers the data for you
    String applicationQuery = 'select id, Name, Ready_To_Join__c from ApplicationForm__c';
    return Database.getQueryLocator(applicationQuery);
}
public void execute(Database.BatchableContext bc, List<ApplicationForm__c> formList){
    // process the data
    for(ApplicationForm__c af : formList){
        totalForms++;
        if(af.Ready_To_Join__c){
            totalConvertedForms++;
        }
    }
}
public void finish(Database.BatchableContext bc){
    // emails ,
```

```

        Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
        // address, subject, content( data to sent to admins)
        mail.setSubject(' Application form and student record data as of today ');
        mail.setPlainTextBody(' Total no of application form records are : '+totalForms+ ' out of
which no of students as per today : '+totalConvertedForms);
        String[] emailAddess = new String[]{'your email address'};
        mail.setToAddresses(emailAddess);
        Messaging.sendEmail(new Messaging.SingleEmailMessage[]{ mail } );
    }
}

```

Day 7:

Here we are going to create the college data table, for this we need to create an apex class, from which we are going to retrieve the data and Html, javascript files for UI .

Create an Apex Class With The Required Functionality

```

public class GetapplicationDetails
{
    @AuraEnabled(cacheable=true)
    public static List<ApplicationForm__c> getapplicationvalues(id
CollegeId)
    {

List<ApplicationForm__c> formlist = [SELECT ID, College_Fees__c, Name,
Date_of_Birth__c, Email__c, Hostel_Fees__c, Student_Name__c FROM
ApplicationForm__c WHERE College__c =:CollegeId];

return formList;
}

}

```

The screenshot shows the Salesforce LWC Editor (Beta) interface. On the left, there's a sidebar with icons for Apex Classes, Lightning Web Components, and other components. The main area shows an Apex class named 'GetapplicationDetails.cls' with the following code:

```
1 public class GetapplicationDetails
2 {
3     @AuraEnabled(cacheable=true)
4     public static List<Application_Form__c> getapplicationvalues(id CollegeId)
5     {
6
7         List<Application_Form__c> formlist = [SELECT ID, College_Fees__c, Name,
8 Date_of_Birth__c, Email__c, Hostel_Fees__c, Student_Name__c FROM
9 Application_Form__c WHERE College__c =:CollegeId];
10
11     return formList;
12 }
```

Below the code editor is a 'Result Panel' showing deployment logs:

Log	Date
collegeManagementApplication.html successfully deployed source	1/21/2022 8:49:09 PM
collegeManagementApplication.js successfully deployed source	1/21/2022 8:49:09 PM
collegeManagementApplication.js-meta.xml successfully deployed source	1/21/2022 8:49:09 PM

Day 8:

Create the component with the following files.

HTML:

```
<template>

    <h1> College and Application form list table </h1>

    <template if:true={recordList}>
        <lightning-datatable
            key-field="id"
            data={recordList}
            show-row-number-column
            hide-checkbox-column
            columns={columnsList}
        >
    </lightning-datatable>
</template>

<template if:true={error}>
```

```

        {error}
    </template>
</template>


```

```

1   <template>
2
3     <h1> College and Application form list table </h1>
4
5     <template if:true={recordList}>
6       <lightning-datable
7         key-field="id"
8         data={recordList}
9         show-row-number-column
10        hide-checkbox-column
11        columns={columnsList}
12      >
13    </lightning-datable>
14  </template>
15
16  <template if:true={error}>
17    {error}
18  </template>
19 </template>

```

Day 9:

```

{label : 'Hostel Fees' , fieldName : 'Hostel_Fees__c', type:'Currency' },
{label : 'StudentName' , fieldName : 'Student_Name__c', type:'text' }
];

```

```

@api recordId;
recordList;
error;

@wire(getapplicationvalues, {CollegId : '$recordId'})
wiredCollegeData({data, error}){
  if(data){
    this.recordList = data;
  }
  else if(error){
    this.error = error;
    this.recordList = undefined;
  }
}
}


```

Salesforce LWC Editor (Beta)

Salesforce LWC Editor

Search Apex or Component...

APEX CLASSES

- ApplicationBatchTest
- applicationschedule
- GetapplicationDetails

LIGHTNING WEB COMPONENTS

- collegeManagementApplication
- collegeManagementApplication
- collegeManagementApplication
- collegeManagementApplication

```
1 import { LightningElement, api, wire } from 'lwc';
2 import getapplicationvalues
3 from '@salesforce/apex/GetapplicationDetails.getapplicationvalues';
4 export default class CollegeDataTable extends LightningElement {
5 columnsList = [
6 {label : 'Application Form' , fieldName : 'Name', type:'text' },
7 {label : 'College Fees' , fieldName : 'College_Fees__c', type:'currency' },
8 {label : 'Date Of Birth' , fieldName : 'Date_Of_Birth__c', type:'date' },
9 {label : 'Email' , fieldName : 'Email__c', type:'email' },
10 {label : 'Hostel Fees' , fieldName : 'Hostel_Fees__c', type:'Currency' },
11 {label : 'StudentName' , fieldName : 'Student_Name__c', type:'text' }
12 ];
13
14 @api recordId;
15 recordList;
16 error;
17
18 @wire(getapplicationvalues, {CollegeId : '$recordId'})
19 wiredCollegeData({data, error}){
20     if(data){
21         this.recordList = data;
22     }
23     else if(error){
24         this.error = error;
25         this.recordList = undefined;
}
}

```

stanley engineering college

Use LWC Standard Deployment? Twitter LinkedIn

Day 10:

```
<?xml version="1.0"?>
<LightningComponentBundle
xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>51.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning__RecordPage</target>
        <target>lightning__AppPage</target>
        <target>lightning__HomePage</target>
    </targets>
</LightningComponentBundle>
```

The screenshot shows the Salesforce LWC Editor (Beta) interface. The left sidebar contains navigation icons and lists APEX CLASSES (ApplicationBatchTest, applicationschedule, GetapplicationDetails) and LIGHTNING WEB COMPONENTS (collegeManagementApplication). The main area displays the code for collegeManagementApplication.js-meta.xml:

```
1  <?xml version="1.0"?>
2  <LightningComponentBundle
3  xmlns="http://soap.force.com/2006/04/metadata">
4      <apiVersion>51.0</apiVersion>
5      <isExposed>true</isExposed>
6      <targets>
7          <target>lightning__RecordPage</target>
8          <target>lightning__AppPage</target>
9          <target>lightning__HomePage</target>
10     </targets>
11 </LightningComponentBundle>
```

The screenshot shows a Salesforce Lightning page for a college record. The top navigation bar includes links for 'Inbox', 'Study', 'SI-7782', 'Home', '(1) Wh...', 'MIT-M...', 'Hands...', 'Home', and a '+' button. The main header features a blue cloud icon, the text 'College Management', and a search bar with placeholder 'Search...'. Below the header, there are tabs for 'Colleges', 'Application Forms', 'Student', and 'Subjects'. On the left, a sidebar displays 'Related' items: 'College Name' (MIT-MUM), 'Record Info', 'College Fees' (\$200,000.00), 'Hostel Fees' (\$30,000.00), 'College Name' (MIT-MUM), 'Email', and 'Capacity Of Students'. The main content area shows the 'Details' tab selected, displaying fields for 'Owner' (Aare Chaitanya, with a profile picture), 'Created By' (Aare Chaitanya, 1/6/2022, 8:30 AM), and 'Last Modified By' (Aare Chaitanya, 1/6/2022, 8:30 AM). A large blue sidebar on the right contains a topographic map graphic. The bottom of the screen shows the Windows taskbar with various pinned icons.