

## Day 1

Topic:Creating Salesforce development account and activating it.

Activity:

Task1:Creating a salesforce developer account

The following steps are involved in creating the salesforce developer account

Creating a developer org in salesforce.

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

2.Click on sign up.

3.On the sign up form, enter the following details :

4.First name & Last name

5.Email

6.Role : Developer

7.Company : College Name

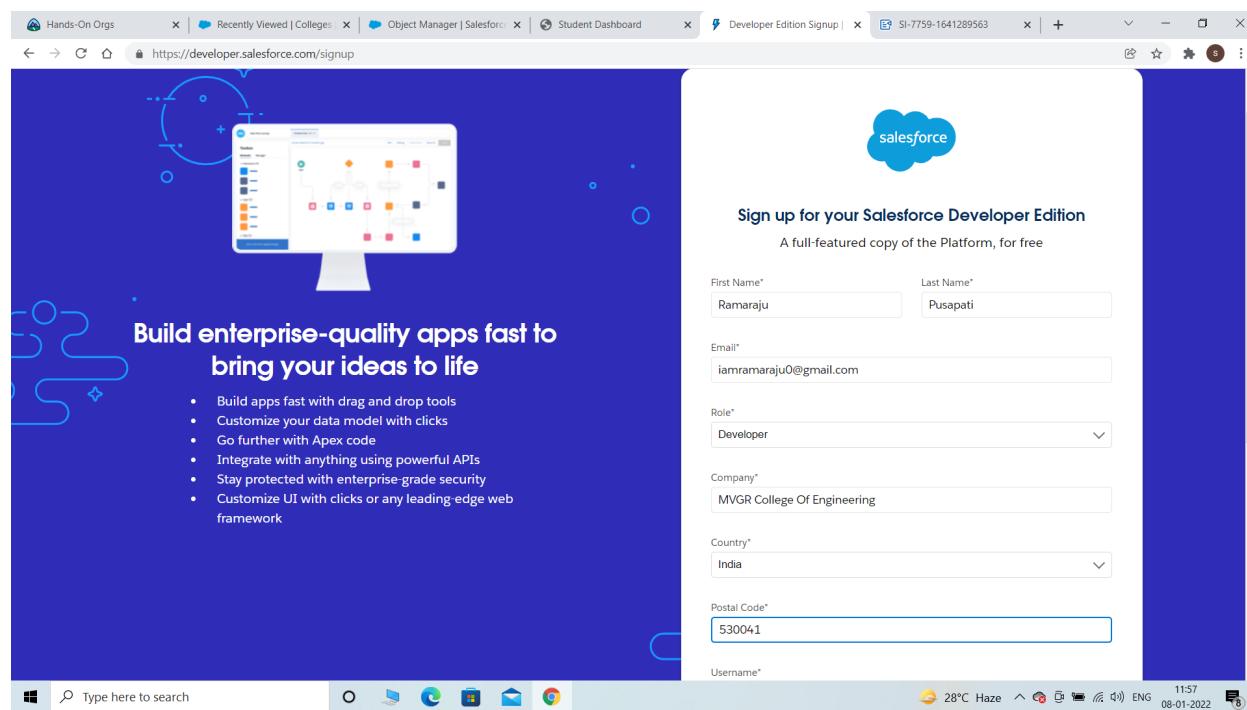
8.County : India

9.Postal Code : pin code

10.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](mailto:username@organization.com)



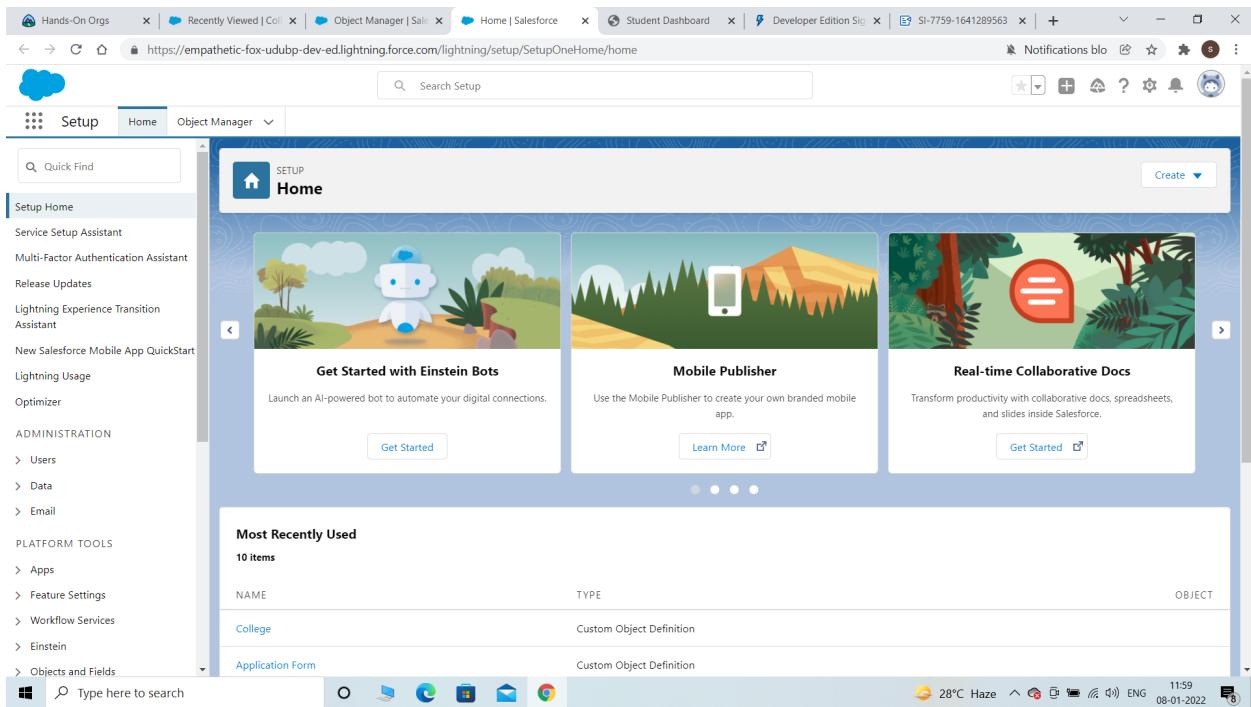
Task2:verifying the account through email verification

You will be receiving a mail from salesforce developer org with a click on button "verify your email" click on it.After clicking on it you will be redirected to the salesforce login page where you will be asked your login credentials to proceed further.

### Task 3 Login into your salesforce developer account

The steps that are needed to follow are:

1. Go to 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:

#### Topic: Costum object creation

Activities:

#### Task1: Create College management Application

The steps involved in this are

1. Click on the settings icon on the salesforce home page this will show two options
2. Click on setup
3. This will open the salesforce home page
4. Now, search for app manager and click on it
5. Click on new lightning app
6. Fill the app details and click next, In the same way fill the app options, navigation items, user profiles and then click on save this creates the new lightning app called College management application

Hands-On Orgs | Recently Viewed | Subjects | College management - Lig | Student Dashboard | Developer Edition Signup | SI-7759-1641289563

Lightning App Builder | App Settings | Pages | College management

### App Settings

#### App Details & Branding

App Options

Utility Items (Desktop Only)

Navigation Items

User Profiles

#### App Details

\* App Name: College management

\* Developer Name: College\_management

Description: This app deals with the management of college system.

#### App Branding

Image:  Upload

Primary Color Hex Value: #007002

Org Theme Options:  Use the app's image and color instead of the org's custom theme

#### App Launcher Preview

Hands-On Orgs | Recently Viewed | Subjects | College management - Lig | Student Dashboard | Developer Edition Signup | SI-7759-1641289563

Lightning App Builder | App Settings | Pages | College management

### App Settings

#### App Options

App Details & Branding

Utility Items (Desktop Only)

Navigation Items

User Profiles

#### Navigation and Form Factor

\* Navigation Style: Standard navigation

\* Supported Form Factors: Desktop and phone

#### Setup and Personalization

Setup Experience:  Setup (full set of Setup options)  Service Setup

App Personalization Settings:

Disable end user personalization of nav items in this app

Disable temporary tabs for items outside of this app

Type here to search

28°C Haze 1608 08-01-2022 ENG

Hands-On Orgs | Recently Viewed | Subjects | College management - Lig | Student Dashboard | Developer Edition Signup | SI-7759-1641289563

Lightning App Builder | App Settings | Pages | College management

### App Settings

#### Utility Items (Desktop Only)

Give your users quick access to productivity tools and add background utility items to your app.

Add Utility Item Utility Bar Alignment Default

The utility bar is a fixed footer that opens components in docked panels. Available only when the app is viewed in Lightning Experience on a desktop.

To enable the utility bar for this app, add a utility item.

Type here to search

Hands-On Orgs | Recently Viewed | Subjects | College management - Lig | Student Dashboard | Developer Edition Signup | SI-7759-1641289563

Lightning App Builder | App Settings | Pages | College management

### App Settings

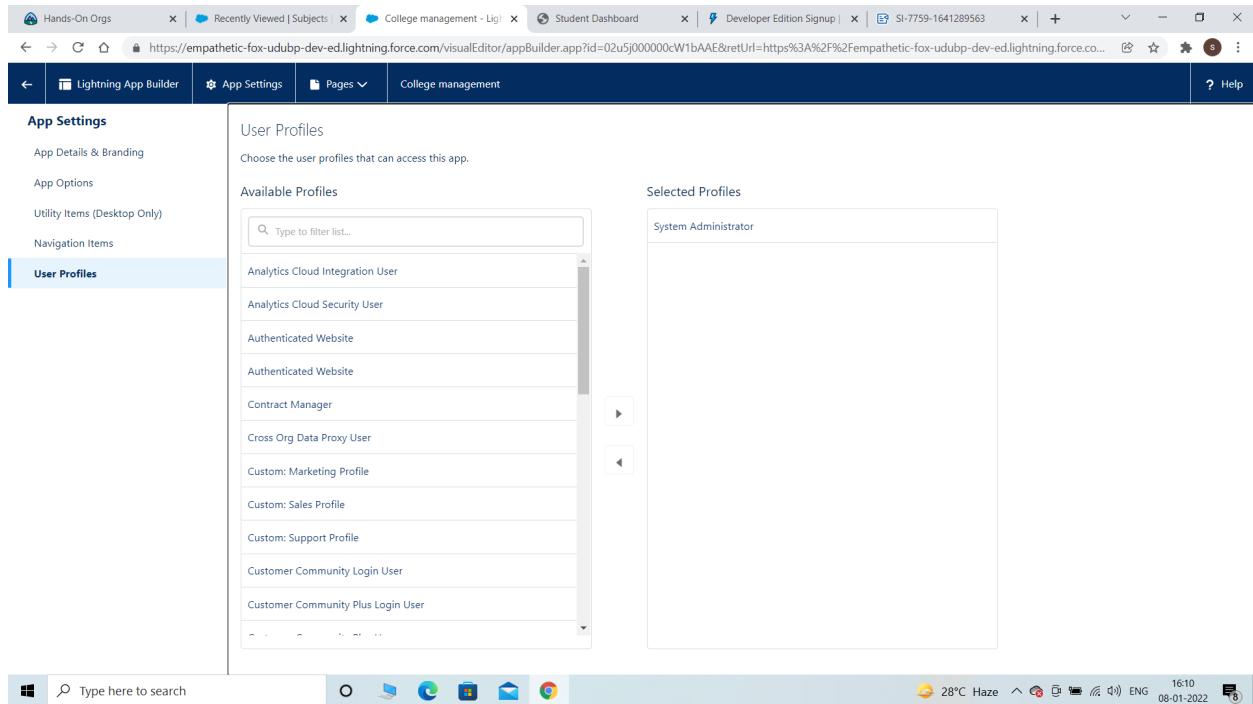
#### Navigation Items

Choose the items to include in the app, and arrange the order in which they appear. Users can personalize the navigation to add or move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.

Available Items Selected Items

Available Items	Selected Items
Accounts	Colleges
Alert Settings	Application Forms
Alternative Payment Methods	Students
App Launcher	Subjects
Approval Requests	
Asset Action Sources	
Asset Actions	
Asset State Periods	
Assets	
Async Operation Logs	
Authorization Form	

Type here to search



## Task2: Create the custom objects for the college management application

The steps involved in creating custom objects are

- 1.Click on setup
- 2.Click on object manager
- 3.Click on create new custom object
- 4.Select the label and fill in the necessary details and click on save

then select a icon for the object and click on save and next to create new objects.

The objects that are needed to be created for college management application are

- 1.College

The screenshot shows the Salesforce Setup interface with the 'Object Manager' tab selected. The main area displays the 'Details' for the 'College' object. The 'API Name' is set to 'College\_\_c'. Under the 'Fields & Relationships' section, the 'Custom' checkbox is checked. In the 'Details' section, the 'Singular Label' is 'College' and the 'Plural Label' is 'Colleges'. The 'Deployment Status' is 'Deployed' and the 'Help Settings' link points to 'Standard salesforce.com Help Window'. The top navigation bar includes tabs for 'Setup', 'Home', and 'Object Manager'. The bottom status bar shows the URL <https://empathetic-fox-udubp-dev.lightning.force.com/lightning/setup/ObjectManager/01I5j000000p5G3/Details/view>, the time 16:12, and the date 08-01-2022.

## 2.Application Form

The screenshot shows the Salesforce Setup interface with the 'Object Manager' tab selected. The main area displays the 'Details' for the 'Application Form' object. The 'API Name' is set to 'Application\_Form\_\_c'. Under the 'Fields & Relationships' section, the 'Custom' checkbox is checked. In the 'Details' section, the 'Singular Label' is 'Application Form' and the 'Plural Label' is 'Application Forms'. The 'Deployment Status' is 'Deployed' and the 'Help Settings' link points to 'Standard salesforce.com Help Window'. The top navigation bar includes tabs for 'Setup', 'Home', and 'Object Manager'. The bottom status bar shows the URL <https://empathetic-fox-udubp-dev.lightning.force.com/lightning/setup/ObjectManager/01I5j000000p5Hu/Details/view>, the time 16:13, and the date 08-01-2022.

## 3.Student

**Student**

**Details**

Description

API Name: Student\_\_c

Singular Label: Student

Plural Label: Students

Enable Reports: ✓

Track Activities: ✓

Track Field History: ✓

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

## 4. Subject

**Subject**

**Details**

Description

API Name: Subject\_\_c

Singular Label: Subject

Plural Label: Subjects

Enable Reports: ✓

Track Activities: ✓

Track Field History: ✓

Deployment Status: Deployed

Help Settings: Standard salesforce.com Help Window

### Task 3: Creating the fields on each of the above objects

The process to create fields is as follows

- 1.click on object manager
- 2.Search the required object in the search bar
- 3.click on the object
- 4.click on fields and relationships

5.Click on new

6.select the data type for the new field and click on next then enter the field label and save  
The objects with their necessary fields that are required to be created for the college management application are:

1.College

The fields that are required in the college object are

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		blr@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 Setup interface for the College object. The left sidebar lists various setup categories. The main content area is titled 'Fields & Relationships' and displays a table of fields. The columns include FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. Fields listed include College\_Email\_c (Picklist), college\_fee\_\_c (Currency(7, 2)), College\_Name\_c (Picklist), Created\_By (Lookup(User)), Hostel\_fee\_\_c (Currency(6, 2)), Last\_Modified\_By (Lookup(User)), No\_of\_Seats\_c (Picklist), Owner (Lookup(User, Group)), and Name (Text(80)).

## 2. Application Form

The fields that are needed to be created on the application form object are:

Create the following fields on the application form object.

Field Name	Data Type	Required	Values
Application Form ID	Autonumber		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		Yes	

Fields & Relationships				
	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address__c	Text(255)		✓
Application Form ID	Application_Form_ID__c	Auto Number		✓
College	College__c	Master-Detail(College)		✓
college fee	college_fee__c	Formula (Currency)		✓
Created By	CreatedById	Lookup(User)		✓
Date of Birth	Date_of_Birth__c	Date		✓
Email	Email__c	Email (Unique)		✓
Form Number	Name	Auto Number		✓
Guardian Name	Guardian_Name__c	Text(30)		✓
Hostel Fees	Hostel_Fees__c	Formula (Currency)		✓
Last Modified By	LastModifiedById	Lookup(User)		✓
Looking For Hostel Stay	Looking_for_Hostel_Stay__c	Checkbox		✓
Phone	Phone__c	Phone		✓
Ready To Join	Ready_to_Join__c	Checkbox		✓
Student Name	Student_Name__c	Text(30)		✓

### 3.Student

The fields that are needed to be created on the student object are:

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	

Fields & Relationships					
	FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Page Layouts	Address	Address_c	Text(255)		
Lightning Record Pages	Application Form	Application_Form_c	Lookup(Application Form)		
Buttons, Links, and Actions	College Name	College_Name_c	Formula (Text)		
Compact Layouts	Created By	CreatedById	Lookup(User)		
Field Sets	Date Of Birth	Date_Of_Birth_c	Date		
Object Limits	Guardian Name	Guardian_Name_c	Text(30)		
Record Types	Last Modified By	LastModifiedById	Lookup(User)		
Related Lookup Filters	Owner	OwnerId	Lookup(User,Group)		
Search Layouts	Phone	Phone_c	Phone		
Search Layouts for Salesforce Classic	Student Name	Student_Name_c	Text(30)		
Restriction Rules	Student Name	Name	Text(80)		
Triggers					
Validation Rules					

#### 4. Subjects

The fields that are required to be created on the subjects object are:

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)		

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		▼
Paper2	Paper2__c	Picklist		▼
Student	Student__c	Lookup(Student)		▼
Subject ID	Subject_ID__c	Auto Number		▼
Subject Name	Name	Text(80)		▼

After creation of the objects and their necessary fields the college management application look as follows:

Subject Name
You haven't viewed any Subjects recently. Try switching list views.

Day 3:

Topic: Adding Business Logic To Application

Task1: Creating global picklist value sets

The process for creating global picklist value sets is as follows:

1. Click on home in setup

2. search for picklist value sets in the search bar and click on it
  3. click on new
  4. enter the label and values to be displayed in the picklist
  5. click on save
  6. while creating the field for picklist select global value sets option and select the required value set
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

The screenshot shows the Salesforce Lightning interface with the following details:

- Page Title:** Picklist Value Sets
- Section:** Global Value Set
- Global Value Set Detail:**
  - Label: College
  - Name: College\_Name
  - Description: (empty)
- Picklist Values Used:**
  - Active and Inactive picklist values: 6 (1,000 max)
- Values Table:**

Action	Values	API Name	Default	Chart Colors	Modified By
Edit   Del   Deactivate	MIT-BLR	MIT-BLR	<input type="checkbox"/>	Assigned dynamically	Ramareju Pusapati 1/6/2022, 9:22 PM
Edit   Del   Deactivate	MIT-MAA	MIT-MAA	<input type="checkbox"/>	Assigned dynamically	Ramareju Pusapati 1/6/2022, 9:22 PM
Edit   Del   Deactivate	MIT-HYD	MIT-HYD	<input type="checkbox"/>	Assigned dynamically	Ramareju Pusapati 1/6/2022, 9:22 PM
Edit   Del   Deactivate	MIT-DEL	MIT-DEL	<input type="checkbox"/>	Assigned dynamically	Ramareju Pusapati 1/6/2022, 9:22 PM
Edit   Del   Deactivate	MIT-MUM	MIT-MUM	<input type="checkbox"/>	Assigned dynamically	Ramareju Pusapati 1/6/2022, 9:22 PM
Edit   Del   Deactivate	MIT-CCU	MIT-CCU	<input type="checkbox"/>	Assigned dynamically	Ramareju Pusapati 1/6/2022, 9:22 PM
- Inactive Values:** No Inactive Values defined.

## b)Paper1

Picklist Value Name	Values
Paper 1	APEX JAVA C C++

The screenshot shows the Salesforce Setup interface with the following details:

- Page Header:** Student Dashboard, SI-7759-1641289563, Hands-On Orgs, Recently Viewed | Colleges, Picklist Value Sets | Salesfo..., Picklist Value Sets | Salesfo...
- Search Bar:** Search Setup
- Setup Navigation:** Home, Object Manager
- Left Sidebar:**
  - Data
  - Picklist Settings
  - State and Country/Territory
  - Picklists** (selected)
  - Objects and Fields
  - Picklist Value Sets** (selected)

Didn't find what you're looking for?  
Try using Global Search.
- Picklist Value Sets Page:**
  - Global Value Set:** Back to List
  - Global Value Set Detail:** Edit, Delete
  - Information:** Label: Paper 1, Name: Paper\_1, Description
  - Picklist Values Used:** Active and inactive picklist values: 4 (1,000 max)
  - Values:**

Action	Values	API Name	Default	Chart Colors	Modified By
Edit   Del   Deactivate	APEX	APEX	<input type="checkbox"/>	Assigned dynamically	Ramaraju Pusapati 1/13/2022, 1:22 AM
Edit   Del   Deactivate	JAVA	JAVA	<input type="checkbox"/>	Assigned dynamically	Ramaraju Pusapati 1/13/2022, 1:22 AM
Edit   Del   Deactivate	C	C	<input type="checkbox"/>	Assigned dynamically	Ramaraju Pusapati 1/13/2022, 1:22 AM
Edit   Del   Deactivate	C++	C++	<input type="checkbox"/>	Assigned dynamically	Ramaraju Pusapati 1/13/2022, 1:22 AM
  - Inactive Values:** No Inactive Values values defined.
  - Fields Where Used:** Fields Label, Object, Data Type, Controlling Field

## c)Paper2

Picklist Value Name	Values
Paper2	MATHEMATICS ENGLISH STATISTICS

The screenshot shows the Salesforce Setup interface. The main page is titled "Picklist Value Sets". On the left, there's a sidebar with "Setup" selected. The main content area shows a "Global Value Set" named "Paper\_2". It has sections for "Information" (Label: Paper\_2, Name: Paper\_2, Description), "Picklist Values Used" (3 values: MATHEMATICS, ENGLISH, STATISTICS), "Values" (a table with columns: Action, Values, API Name, Default, Chart Colors, Modified By), "Inactive Values" (empty), and "Fields Where Used" (empty). The bottom of the screen shows a taskbar with various icons and a system status bar indicating "29°C Rain showers" and the date "13-01-2022".

## Task2:Creating Field Dependencies

The process to create field dependencies is as follows

- 1.Go to object manager
- 2.click on college object
- 3.click on fields and relationships
- 4.click on field dependencies
- 5.click on new

6.select the controlling field and the dependent field

7.select the values to be included and save it

The field dependencies that we need to create are:

- 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.

### Task3:Creating validation rules on objects

The steps required to follow for creation of validation rules are:

1.Click on object manager

2.go to the required object

3.click on validation rules

4.click on new

5.add the rule and save it

The objects on which the validation rules have to be created are:

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

TEXT(College\_Name\_\_c) <> Name.

The screenshot shows the Salesforce Object Manager interface. On the left, a sidebar lists various setup options like Details, Fields & Relationships, Page Layouts, etc. The main area is titled 'College Validation Rule' under 'Validation Rule Edit'. It shows a 'Rule Name' field with 'College\_Name\_similar', an 'Active' checkbox checked, and a 'Description' field. Below this is the 'Error Condition Formula' section, which contains the formula 'TEXT(College\_Name\_\_c) <> Name'. A dropdown menu labeled 'Functions' is open, showing options like ABS, ADDMONTHS, AND, BEGINS, BLANKVALUE, and BR. A 'Quick Tips' box is visible on the right. At the bottom, there's an 'Error Message' section and a 'Check Syntax' button. The top navigation bar shows tabs for Student Dashboard, SL-7759-1641289563, Hands-On Orgs, All Application Forms, Picklist Value Sets, College, and a search bar. The status bar at the bottom indicates 27°C Light rain, 15:46, and 13-01-2022.

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\_\_c == true,

OR(

ISCHANGED( Address\_\_c ),

ISCHANGED( College\_\_c ),

ISCHANGED( DateOfBirth\_\_c ),

ISCHANGED( Email\_\_c ),

ISCHANGED( GuardianName\_\_c ),

ISCHANGED( Phone\_\_c )

)

)

**Application Form Validation Rule**

Validation Rule Edit

Rule Name: No\_modification\_of\_Application

Active:

Description:

Error Condition Formula

Example: Discount\_Percent\_c < 0.30 | More Examples...  
Display an error if Discount is more than 30%.

If this formula expression is true, display the text defined in the Error Message area.

Insert Field | Insert Operator | Functions

ABS  
ADDMONTHS  
AND  
BEGINS  
BLANKVALUE  
BR  
Insert Selected Function  
ABS(number)  
Returns the absolute value of a number, a number without its sign  
Help on this function

Check Syntax

Error Message

## Task4: Process Automation

Steps required for process Automation are:

- 1.click on the home
- 2.search for process builder
- 3.click on new
- 4.enter the necessary fields
- 5.click on activate then save

Process Builder - Application Form Process master

Action Name: create a student record

Record Type: Student

Set Field Values

Field *	Type *	Value *
Address	Field Reference	[Application_Form_c].Add... Q
Date Of Birth	Field Reference	[Application_Form_c].Dat... Q
Guardian Name	Field Reference	[Application_Form_c].Gua... Q
Phone	Field Reference	[Application_Form_c].Pho... Q
Application Form	Field Reference	[Application_Form_c].Id Q
Student Name	Field Reference	[Application_Form_c].Stu... Q

## Task5:Create the student record using flow

The steps required to build the flow are as follows:

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.

The screenshot shows the Flow Builder interface with the following details:

- Toolbox:** On the left, under "Manager", there is a search bar and a "New Resource" button. Under "RESOURCES", there is a section for "Record (Single) Variables" containing "studentVar". Under "ELEMENTS", there is a section for "Assignments" containing "Assign\_student\_record\_data" and a section for "Create Records" containing "Create\_Student\_Record".
- Flow Diagram:** The main area shows a "Record-Triggered Flow" starting with a "Run Immediately" step, followed by an "Assignment" step labeled "Assign student record data", then a "Create Student Record" step, and finally an "End" step.
- Edit Assignment Dialog:** A modal dialog titled "Edit Assignment" is open, showing the configuration for the "Assign\_student\_record\_data" assignment. It lists seven variable assignments:
  - Variable: \$studentVar > Address, Operator: Equals, Value: \$Record > Address
  - Variable: \$studentVar > Application Form, Operator: Equals, Value: \$Record > Application Form ID
  - Variable: \$studentVar > College Name, Operator: Equals, Value: \$Record > College
  - Variable: \$studentVar > Date Of Birth, Operator: Equals, Value: \$Record > date of Birth
  - Variable: \$studentVar > Guardian Name, Operator: Equals, Value: \$Record > Guardian Name
  - Variable: \$studentVar > Phone, Operator: Equals, Value: \$Record > Phone
  - Variable: \$studentVar > Student Name, Operator: Equals, Value: \$Record > Student Name

Day 6

Topic:Batch Apex

Batch Apex is used to run large jobs (think thousands or millions of records!) that would exceed

normal processing limits. Using Batch Apex, you can process records asynchronously in batches (hence the name, “Batch Apex”) to stay within platform limits. If you have a lot of records to process, for example, data cleansing or archiving, Batch Apex is probably your best solution.

## Syntax Of BatchApex

```
public class MyBatchClass implements Database.Batchable<sObject> {
    public (Database.QueryLocator | Iterable<sObject>) start(Database.BatchableContext bc) {
        // collect the batches of records or objects to be passed to execute
    }
    public void execute(Database.BatchableContext bc, List<P> records){
        // process each batch of records
    }
    public void finish(Database.BatchableContext bc){
        // execute any post-processing operations
    }
}
```

Task1:Create a batch apex for Application form

For this you need to create a apex class and do the following

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

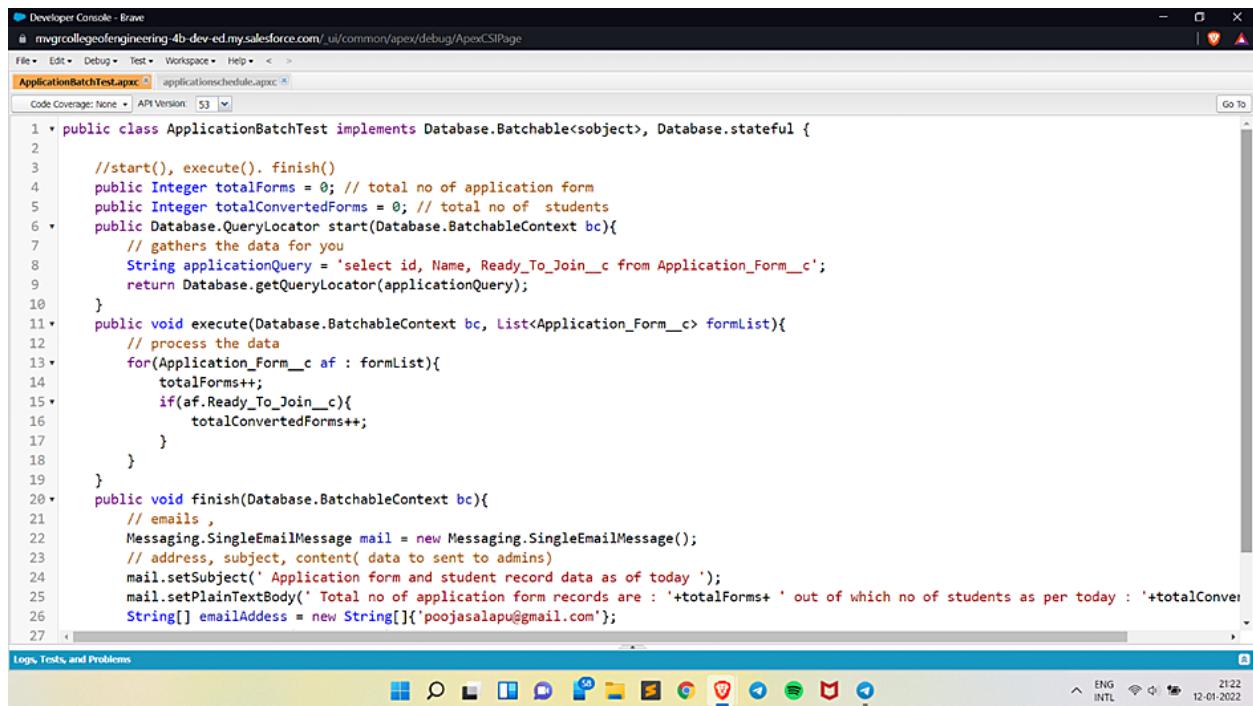
```
Public class ApplicationBatchTest implements Database.Batchable<sObject>{

    //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 });
}
}

```



The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Brave" and the URL is "mvgcollegeofengineering-4b-dev-ed.my.salesforce.com/.ui/common/apex/debug/ApexCSIPage". The tab bar has "ApplicationBatchTest.apxc" selected. The code editor displays the following Apex class:

```

1 * public class ApplicationBatchTest implements Database.Batchable<sobject>, Database.stateful {
2
3     //start(), execute(). finish()
4     public Integer totalForms = 0; // total no of application form
5     public Integer totalConvertedForms = 0; // total no of students
6     public Database.QueryLocator start(Database.BatchableContext bc){
7         // gathers the data for you
8         String applicationQuery = 'select id, Name, Ready_To_Join__c from Application_Form__c';
9         return Database.getQueryLocator(applicationQuery);
10    }
11    public void execute(Database.BatchableContext bc, List<Application_Form__c> formList){
12        // process the data
13        for(Application_Form__c af : formList){
14            totalForms++;
15            if(af.Ready_To_Join__c){
16                totalConvertedForms++;
17            }
18        }
19    }
20    public void finish(Database.BatchableContext bc){
21        // emails ,
22        Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
23        // address, subject, content( data to sent to admins)
24        mail.setSubject(' Application form and student record data as of today ');
25        mail.setPlainTextBody(' Total no of application form records are : '+totalForms+ ' out of which no of students as per today : '+totalConvertedForms);
26        String[] emailAddess = new String[]{'poojasalapu@gmail.com'};
27    }
}

```

The status bar at the bottom shows "Logs, Tests, and Problems", "ENG INTL", "21:22", and "12-01-2022".

Developer Console - Brave  
 mvrgcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File Edit Debug Test Workspace Help < >

ApplicationBatchTest.apxc applicationschedule.apxc

Code Coverage: None API Version: 53 Go To

```

5 lic Integer totalConvertedForms = 0; // total no of students
6 *lic Database.QueryLocator start(Database.BatchableContext bc){
7 // gathers the data for you
8 String applicationQuery = 'select id, Name, Ready_To_Join__c from Application_Form__c';
9 return Database.getQueryLocator(applicationQuery);
10
11*lic void execute(Database.BatchableContext bc, List<Application_Form__c> formList){
12 // process the data
13 for(Application_Form__c af : formList){
14     totalForms++;
15     if(af.Ready_To_Join__c){
16         totalConvertedForms++;
17     }
18 }
19
20*lic void finish(Database.BatchableContext bc){
21 // emails ,
22 Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
23 // address, subject, content( data to sent to admins)
24 mail.setSubject(' Application form and student record data as of today ');
25 mail.setPlainTextBody(' Total no of application form records are : '+totalForms+ ' out of which no of students as per today : '+totalConvertedForm
26 String[] emailaddress = new String[]{poojasalapu@gmail.com};
27 mail.setToAddresses(emailaddress);
28 Messaging.sendEmail(new Messaging.SingleEmailMessage[]{ mail } );
29
30

```

Logs, Tests, and Problems

Developer Console - Brave  
 mvrgcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File Edit Debug Test Workspace Help < >

ApplicationBatchTest.apxc applicationschedule.apxc

Code Coverage: None API Version: 53 Go To

```

1 * public class ApplicationBatchTest implements Database.Batchable<sobject>, Database.stateful {
2
3     //start(), execute(). finish()
4     public Integer totalForms = 0; // total no of application form
5     public Integer totalConvertedForms = 0; // total no of students
6     public Database.QueryLocator start(Database.BatchableContext bc){
7         // gathers the data for you
8         String applicationQuery = 'select id, Name, Ready_To_Join__c from Application_Form__c';
9         return Database.getQueryLocator(applicationQuery);
10    }
11   public void execute(Database.BatchableContext bc, List<Application_Form__c> formList){
12     // process the data
13     for(Application_Form__c af : formList){
14         totalForms++;
15         if(af.Ready_To_Join__c){
16             totalConvertedForms++;
17         }
18     }
19   }
20   public void finish(Database.BatchableContext bc){
21     // emails ,
22     Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
23     // address, subject, content( data to sent to admins)
24     mail.setSubject(' Application form and student record data as of today ');
25     mail.setPlainTextBody(' Total no of application form records are : '+totalForms+ ' out of which no of students as per today : '+totalConve
26     String[] emailAddress = new String[]{poojasalapu@gmail.com};
27

```

Logs, Tests, and Problems

```
Developer Console - Brave
mvrgcollegeofengineering-4b-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help < >
ApplicationBatchTest.apc applicationschedule.apc
Code Coverage: None API Version: 53 Go To
5 lic Integer totalConvertedForms = 0; // total no of students
6 *lic Database.QueryLocator start(Database.BatchableContext bc){
7 // gathers the data for you
8 String applicationQuery = 'select id, Name, Ready_To_Join__c from Application_Form__c';
9 return Database.getQueryLocator(applicationQuery);
10
11 *lic void execute(Database.BatchableContext bc, List<Application_Form__c> formList){
12 // process the data
13 *for(Application_Form__c af : formList){
14     totalForms++;
15 *     if(af.Ready_To_Join__c){
16         totalConvertedForms++;
17     }
18 }
19
20 *lic void finish(Database.BatchableContext bc){
21 // emails ,
22 Messaging.SingleEmailMessage mail = new Messaging.SingleEmailMessage();
23 // address, subject, content( data to sent to admins)
24 mail.setSubject(' Application form and student record data as of today ');
25 mail.setPlainTextBody(' Total no of application form records are : '+totalForms+ ' out of which no of students as per today : '+totalConvertedForm
26 String[] emailaddress = new String[]{'poojasalapu@gmail.com'};
27 mail.setToAddresses(emailaddress);
28 Messaging.sendEmail(new Messaging.SingleEmailMessage[]{ mail });
29
30
Logs, Tests, and Problems
21:22 ENG INTL 12-01-2022
```

## Task2:Create a scheduler class

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

```
public class applicationschedule implements Schedulable{
    public void execute(SchedulableContext sc){
        ApplicationBatchTest abt = new ApplicationBatchTest();
        Database.executeBatch(abt, 400); // 200 to 200
    }
}
```

From setup search for apex class and click on schedule jobs and fill the details as per your requirements.

Developer Console - Brave

mvgcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File Edit Debug Test Workspace Help < >

ApplicationBatchTest.apxc applicationschedule.apxc

Code Coverage: None API Version: 53 Go To

```
1 public class applicationschedule implements Schedulable{  
2  
3  
4  
5     public void execute(SchedulableContext sc){  
6  
7         ApplicationBatchTest abt = new ApplicationBatchTest();  
8  
9         Database.executeBatch(abt, 400); // 200 to 2000  
10  
11     }  
12  
13 }  
14  
15 }
```

Logs, Tests, and Problems

Execute Anonymous - Brave

mvgcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexExecAnon

```
1 ApplicationBatchTest abt=new ApplicationBatchTest();  
2 Database.executeBatch(abt, 400);  
3  
4  
5 String crn='20 10 23 ? 1 1';  
6 System.schedule('cronJob1', crn, new applicationschedule());
```

Open Log Execute Execute Highlighted

WhatsApp Student Dashboard Triggers | SmartBridge SI-7885-1641227630 Apex Jobs | Salesforce mvgrcollegeengineering-4b-dev-ed.lightning.force.com/lightning/setup/AsyncApexJobs/home

Action	Submitted Date	Job Type	Status	Status Detail	Total Batches	Batches Processed	Failures	Submitted By	Completion Date	Apex Class	Apex Method	Apex Job ID
1/12/2022, 2:30 AM	Batch Apex	Completed			1	1	0	Salapu, Pooja Madhurima	1/12/2022, 2:30 AM	ApplicationBatchTest		7075g00001jTdSE
1/12/2022, 2:30 AM	Scheduled Apex	Queued			0	0	0	Salapu, Pooja Madhurima		applicationschedule		7075g00001jTdSG
1/12/2022, 2:15 AM	Scheduled Apex	Aborted			0	0	0	Salapu, Pooja Madhurima		applicationschedule		7075g00001jTcZB
1/12/2022, 2:12 AM	Batch Apex	Completed			1	1	0	Salapu, Pooja Madhurima	1/12/2022, 2:12 AM	ApplicationBatchTest		7075g00001jTdF6
1/12/2022, 2:09 AM	Batch Apex	Completed		First error: SendEmail failed. First exception on row 0; first error: INVALID_ID: Invalid ID. ID is invalid or you do not have access to the record.: [toAddresses, your]	1	1	0	Salapu, Pooja Madhurima	1/12/2022, 2:09 AM	ApplicationBatchTest		7075g00001jTcbu

WhatsApp Student Dashboard Triggers | SmartBridge SI-7885-1641227630 Scheduled Jobs | Salesforce mvgrcollegeengineering-4b-dev-ed.lightning.force.com/lightning/setup/ScheduledJobs/home

Action	Job Name	Submitted By	Submitted	Started	Next Scheduled Run	Type
Del	cronJob1	Salapu, Pooja Madhurima	1/12/2022, 2:30 AM		1/15/2022, 11:10 PM	Scheduled Apex
Del	Analytics Data Loader Job for Org 00D5g000004zBVk	User_Integration	6/12/2021, 8:13 PM	1/11/2022, 8:37 PM	1/12/2022, 8:37 PM	Autonomous Data Loader Job

Developer Console - Brave  
mvrgcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < ▾

applicationFormTrigger.apxc formhelper.apxc formhelper\_TestLapoc ▾

Code Coverage: None ▾ API Version: 53 ▾

```
1 * public class formhelper {
2     public static void getoldNewValues(Map<id, Application_Form__c> formoldMap, List<Application_Form__c> formlist)
3     {
4         for(Application_Form__c fm:formlist)
5         {
6             Application_Form__c oldFormRecord = formoldMap.get(fm.id);
7             System.debug('Old value : '+oldFormRecord.Student_Name__c);
8             System.debug('New Values : '+fm.Student_Name__c + 'Old Values : '+formoldMap.get(fm.id).Student_Name__c);
9             if(fm.Student_Name__c!=formoldMap.get(fm.id).Student_Name__c)
10            {
11            }
12        }
13    }
14    public static void preventRecordDeletion(List<Application_Form__c> formlist)
15    {
16        Set<id> formIdSet=new Set<id>();
17        Map<id,Students__c> formidStudentRecMap=new Map<id,Students__c>();
18        for(Application_Form__c fm:formlist)
19        {
20            formIdSet.add(fm.id);
21        }
22        if(formIdSet!=null && formIdSet.size()>0)
23        {
24            List<Students__c> stdList=[select id,Application_Form__c from Students__c where Application_Form__c IN :formIdSet];
25            if(stdList.size()>0)
26            {
27
```

Logs, Tests, and Problems

Developer Console - Brave  
mvrgcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexCSIPage

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < ▾

applicationFormTrigger.apxc formhelper.apxc formhelper\_TestLapoc ▾

Code Coverage: None ▾ API Version: 53 ▾

```
27    {
28        for(Students__c std:stdList){
29            formidStudentRecMap.put(std.Application_Form__c,std);
30        }
31    }
32    if(formidStudentRecMap!=null && formidStudentRecMap.values().size()>0)
33    {
34        for(Application_Form__c fm:formlist){
35            if(formidStudentRecMap.containsKey(fm.id) && formidStudentRecMap.get(fm.id)!=null){
36                fm.addError(System.label.Application_form_deletion_error_messages);
37            }
38        }
39    }
40 }
41
42 public static void preventMultipleStudentRecordCreation(List<Application_Form__c> formlist) {
43     Set<id> formIdSet=new Set<id>();
44     Map<id,Students__c> formidStudentRecMap=new Map<id,Students__c>();
45     List<Subjects__c> stdlist2 = new List<Subjects__c>();
46     for(Application_Form__c fm:formlist)
47     {
48         formIdSet.add(fm.id);
49     }
50     if(formIdSet!=null && formIdSet.size()>0)
51     {
52         List<Students__c> stdList=[select id, Application_Form__c from Students__c where Application_Form__c IN :formIdSet];
53         if(stdList.size()>0)
```

```

2 • public class formhelper_Test {
3   v testMethod static void getoldNewValuesTest(){
4     College_c clg=new College_c();
5     clg.College_names_c = 'MIT - BLR';
6     clg.Email_c='mit@blr.com';
7     clg.College_Fees_c=52877;
8     clg.Name='MIT - BLR';
9     insert clg;
10    Application_Form_c af=new Application_Form_c();
11    af.College_c=clg.id;
12    af.date_of_Birth_c=System.today()-720;
13    af.Guardian_Name_c='Guardian';
14    af.Email_c='henry@herry.com';
15    af.Student_Name_c='Henry Henry';
16    af.Phone_c='1234567891';
17    af.Address_c='Chennai';
18    insert af;
19
20    //List<Students_c> stdList=[select id, Application_Form_c from Students_c where Application_Form_c =: af.id];
21    //System.debug('List of Students : '+stdList);
22    af.Student_Name_c='Henry porter';
23    af.Ready_To_Join_c=true;
24    update af;
25
26
27    List<Students_c> stdList=[select id, Application_Form_c from Students_c where Application_Form_c =: af.id];
28    System.debug('List of Students : '+stdList);

```

Day5:

Topic:Lightening web components

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 .

Task1:Create college data table component(APEX CLASS)

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;
    }
}

```

```

11 < Apex > utilityControl
12 }
13 @AuraEnabled(cacheable=true)
14 public static List<formWrapper> getApplicationDetails(id collegeId){
15     List<Application_Form__c> formList=[select id, College_Fees__c, Email__c, Date_of_Birth__c, Address__c, Guardian_Name__c, Hostel_Fees__c, collegeName__c from Application_Form__c];
16     List<formWrapper> wrapperList=new List<formWrapper>();
17     for(Application_Form__c af:formList){
18         formWrapper wrap=new formWrapper();
19         wrap.Name=af.Name;
20         wrap.Student_Name=af.Student_Name__c;
21         wrap.Guardian_Name=af.Guardian_Name__c;
22         wrap.College_Fees=af.College_Fees__c;
23         wrap.Hostel_fees=af.Hostel_Fees__c;
24         wrap.Address=af.Address__c;
25         wrap.Email=af.Email__c;
26         wrap.collegeName=af.college__r.College_Names__c;
27         wrap.formNameUrl='/'+af.id;
28         wrap.DOB=af.Date_of_Birth__c;
29         wrapperList.add(wrap);
30     }
31     return wrapperList;
32 }
33
34 public class formWrapper{
35     @AuraEnabled public String Name {get;set;}
36     @AuraEnabled public String student_Name {get;set;}
37     @AuraEnabled public String Guardian_Name {get;set;}
38     @AuraEnabled public Decimal College_fees {get;set;}
39     @AuraEnabled public Decimal Hostel_fees {get;set;}
40     @AuraEnabled public String Address {get;set;}
41     @AuraEnabled public String Email {get;set;}
42     @AuraEnabled public String collegeName {get;set;}
43     @AuraEnabled public String formNameUrl {get;set;}
44     @AuraEnabled public Date Dob {get;set;}
45 }
46 }

```

## Task2:Create college data table component(HTML CLASS)

Create the component with the following files.

HTML:

<template>

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

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

</lightning-datable>

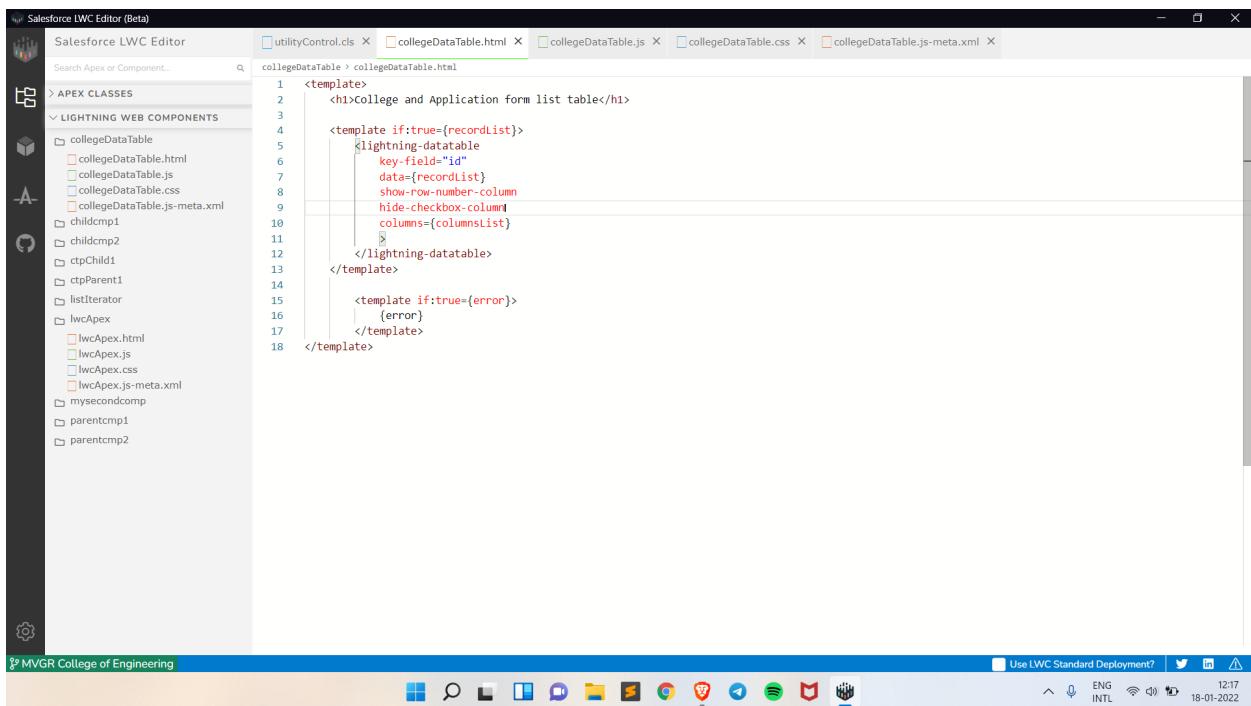
</template>

<template if:true={error}>

{error}

</template>

</template>



### Task3:

Topic:Create college data table component(SCRIPT CLASS)

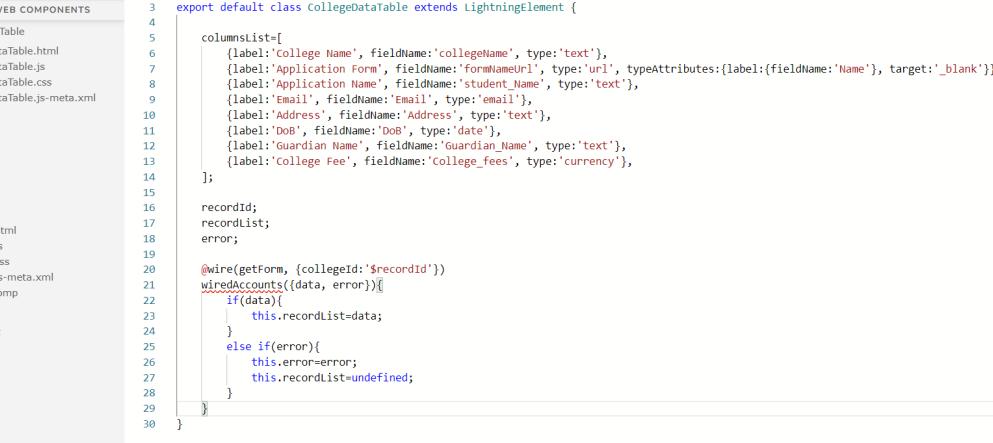
After the creation of the Html file create the following js file

```
import { LightningElement, api, wire } from 'lwc';
import getapplicationvalues
from'@salesforce/apex/GetapplicationDetails.getapplicationvalues';
export default class CollegeDataTable extends LightningElement {
columnsList = [
{label : 'Application Form' , fieldName : 'Name', type:'text' },
{label : 'College Fees' , fieldName : 'College_Fees__c', type:'currency' },
{label : 'Date Of Birth' , fieldName : 'Date_Of_Birth__c', type:'date' },
{label : 'Email' , fieldName : 'Email__c', type:'email' },
{label : 'Hostel Fees' , fieldName : 'Hostel_Fees__c', type:'Currency' },
{label : 'StudentName' , fieldName : 'Student_Name__c', type:'text' }
];

@api recordId;
recordList;
error;

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

```
        }  
    }  
    else if(error){  
        this.error = error;  
        this.recordList = undefined;  
    }  
}
```



The screenshot shows the Salesforce LWC Editor interface. The left sidebar lists various components and classes, including 'collegeDataTable' under 'LIGHTNING WEB COMPONENTS'. The main area displays the source code for 'collegeDataTable.js'.

```
1 import { LightningElement, api, wire } from 'lwc';
2 import getForm from '@salesforce/apex/utilityControl.getApplicationDetails';
3 export default class CollegedataTable extends LightningElement {
4
5     columnslist=[
6         {label:'College Name', fieldName:'collegeName', type:'text'},
7         {label:'Application Form', fieldName:'formNameUrl', type:'url', typeAttributes:{label:{fieldName:'Name'}, target:'_blank'}},
8         {label:'Application Name', fieldName:'student_Name', type:'text'},
9         {label:'Email', fieldName:'Email', type:'email'},
10        {label:'Address', fieldName:'Address', type:'text'},
11        {label:'DoB', fieldName:'DoB', type:'date'},
12        {label:'Guardian Name', fieldName:'Guardian_Name', type:'text'},
13        {label:'College Fee', fieldName:'College_fees', type:'currency'},
14    ];
15
16    recordId;
17    recordList;
18    error;
19
20    @wire(getForm, {collegeId:$recordId})
21    wiredAccounts({data, error}){
22        if(data){
23            this.recordList=data;
24        }
25        else if(error){
26            this.error=error;
27            this.recordList=undefined;
28        }
29    }
30 }
```

The screenshot shows the Salesforce LWC Editor (Beta) interface. The left sidebar displays a tree view of components and files, including APEX CLASSES and LIGHTNING WEB COMPONENTS like collegeDataTable, childcmp1, childcmp2, ctpChild1, ctpParent1, listIterator, lwcApex, mysecondcomp, parentcmp1, and parentcmp2. The right pane shows the code editor for collegeDataTable.css, which contains the following CSS:

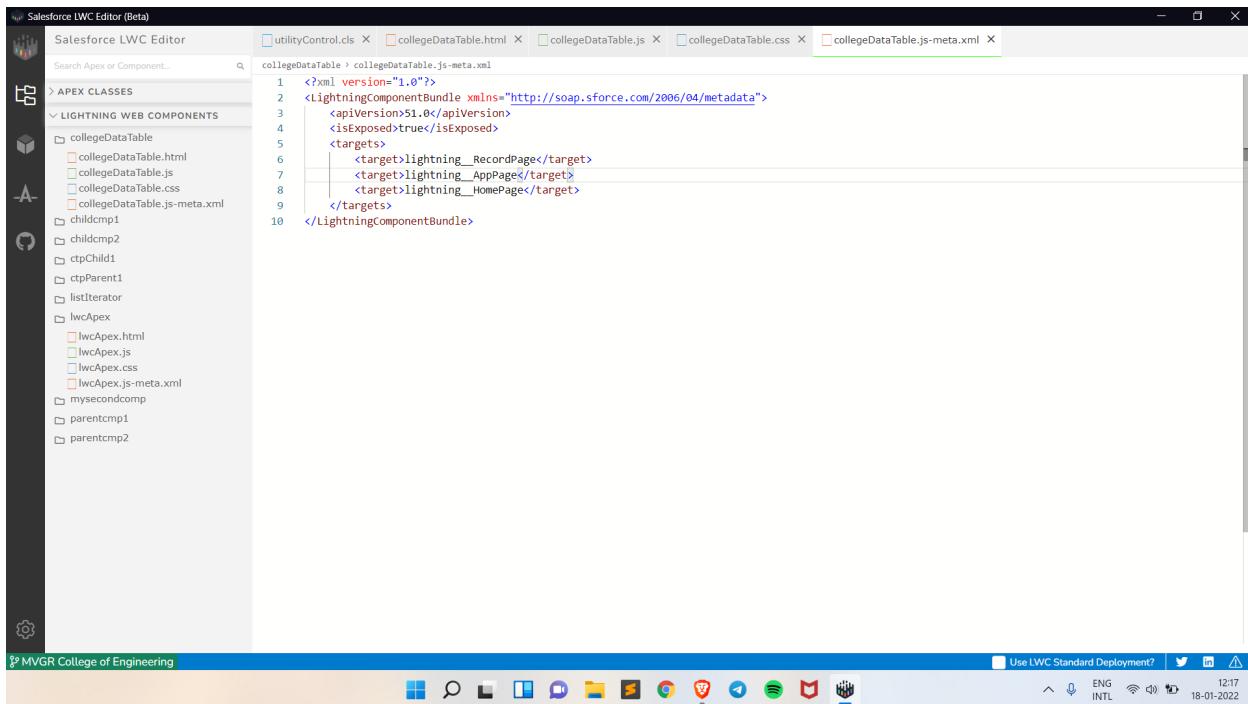
```
1 .dummy{  
2   color : black;  
3 }
```

#### Task4:

Topic:Create college data table component(META CLASS)

Change the Meta File as follows:

```
<?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>
```



#### CONCLUSION:

In this project we got to learn the various steps of a college management application development with the help of various milestones like creating objects creating fields on the objects, defining the dependencies between the fields ,creating validation rules,Custom object creation,adding business logic to the application,creating apex class the lightening web components.

