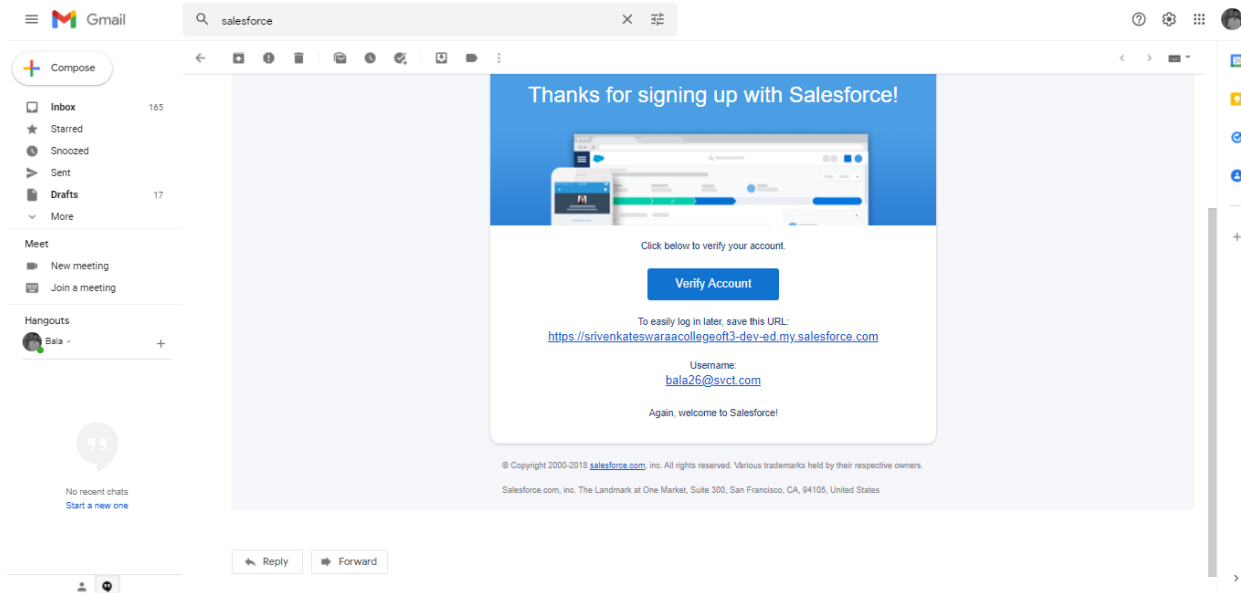
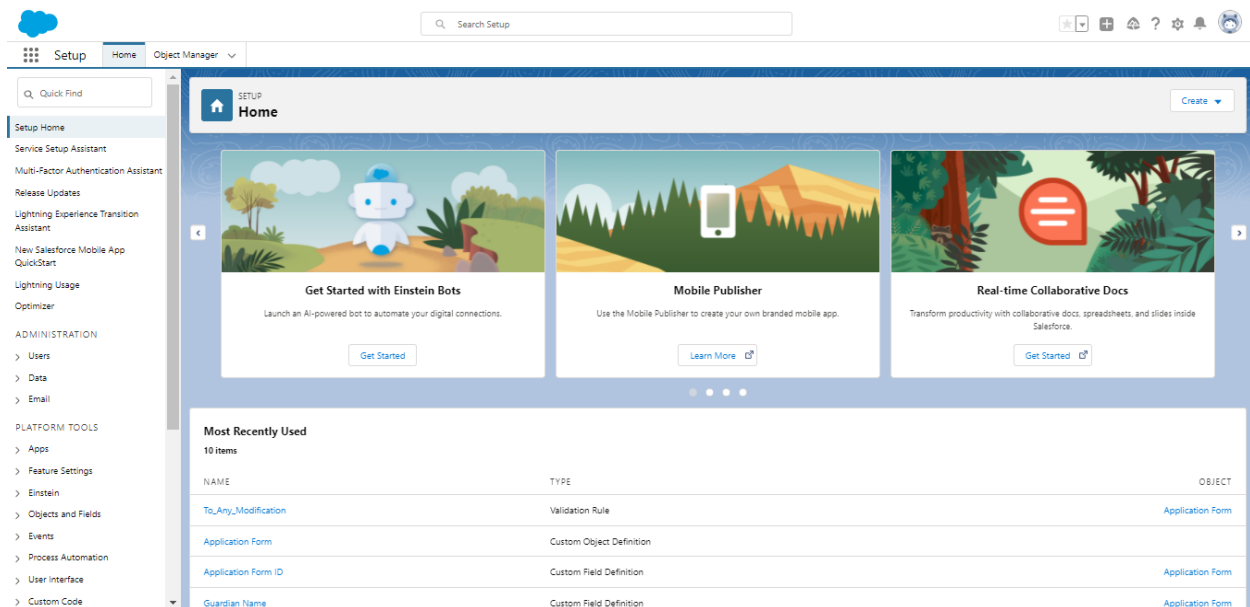


# Task 1 : Create Your Salesforce Developer Org To Get Started

## Account Activation

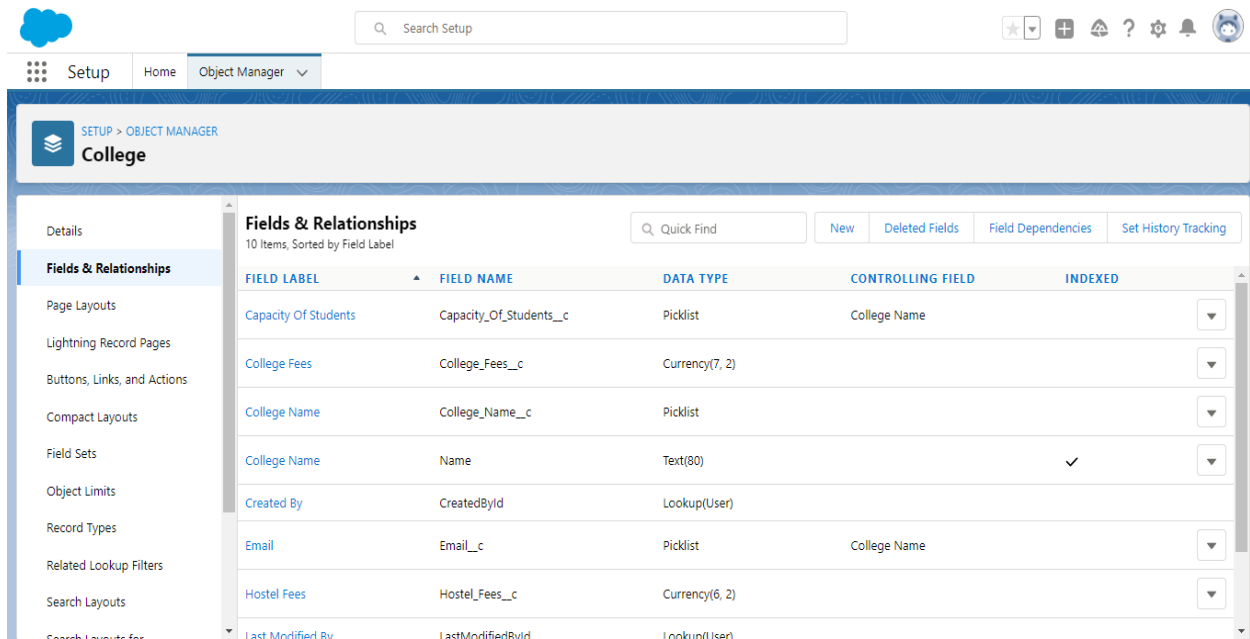


## Salesforce Login



## Task 2 : Create Custom Object

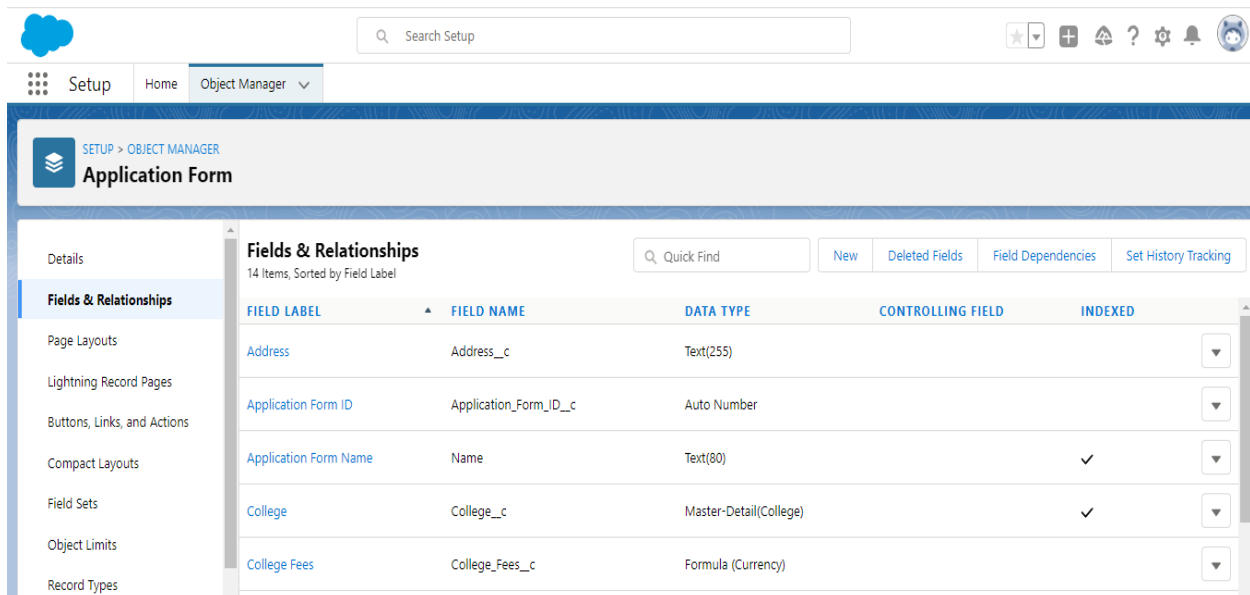
### Create Fields On College Object



The screenshot shows the Salesforce Setup interface for the 'College' object. The 'Fields & Relationships' section is active, displaying a list of 10 fields. The fields are sorted by Field Label. The table includes columns for Field Label, Field Name, Data Type, Controlling Field, and Indexed status.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Capacity of Students	Capacity_Of_Students__c	Picklist	College Name	
College Fees	College_Fees__c	Currency(7, 2)		
College Name	College_Name__c	Picklist		
College Name	Name	Text(80)		✓
Created By	CreatedById	Lookup(User)		
Email	Email__c	Picklist	College Name	
Hostel Fees	Hostel_Fees__c	Currency(6, 2)		
Last Modified By	LastModifiedById	Lookup(User)		

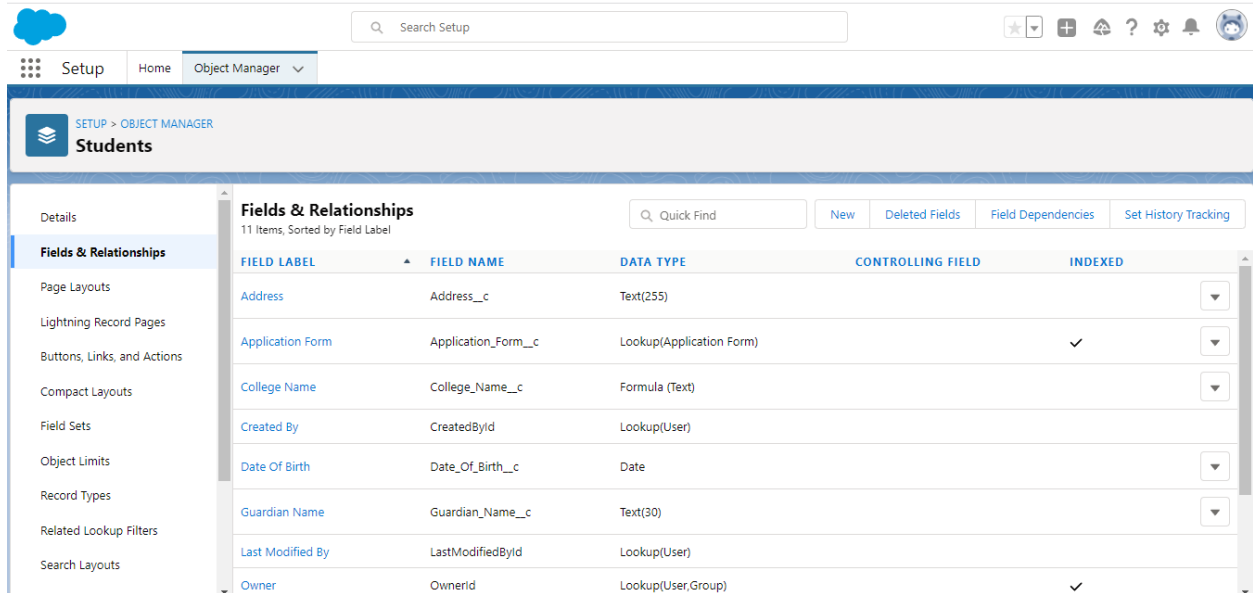
### Create Fields On Application Form Object



The screenshot shows the Salesforce Setup interface for the 'Application Form' object. The 'Fields & Relationships' section is active, displaying a list of 14 fields. The fields are sorted by Field Label. The table includes columns for Field Label, Field Name, Data Type, Controlling Field, and Indexed status.

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)		✓
College Fees	College_Fees__c	Formula (Currency)		

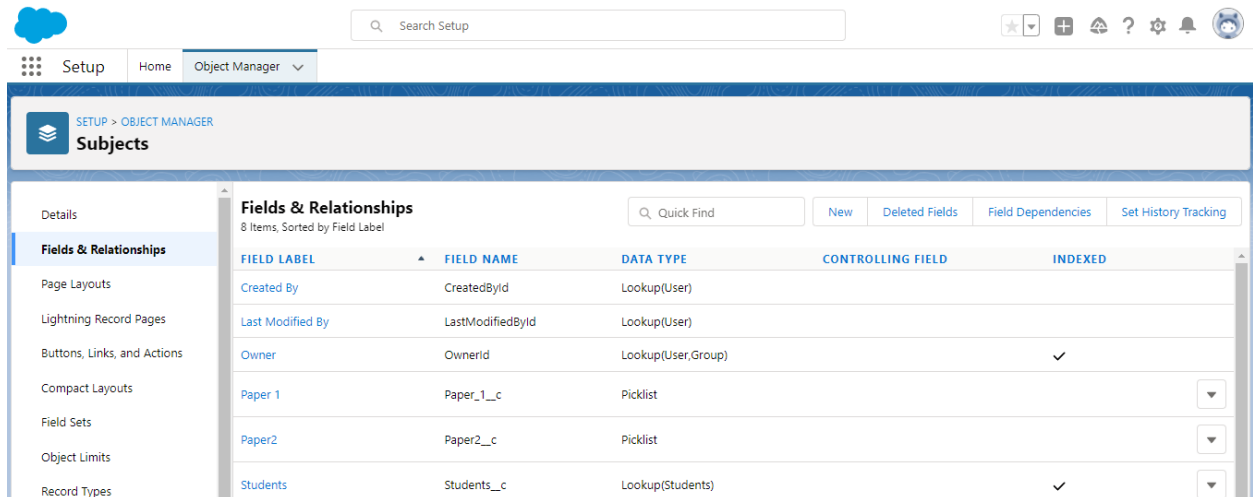
## Create Fields On Student Object



The screenshot shows the Salesforce Setup interface for the 'Students' object. The 'Fields & Relationships' section is active, displaying a list of 11 fields. The left sidebar contains navigation links: Details, 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 top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The top right corner has a search bar and various utility icons.

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	CreatedById	Lookup(User)		
Date Of Birth	Date_Of_Birth__c	Date		
Guardian Name	Guardian_Name__c	Text(30)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓

## Create Fields On Subject Object

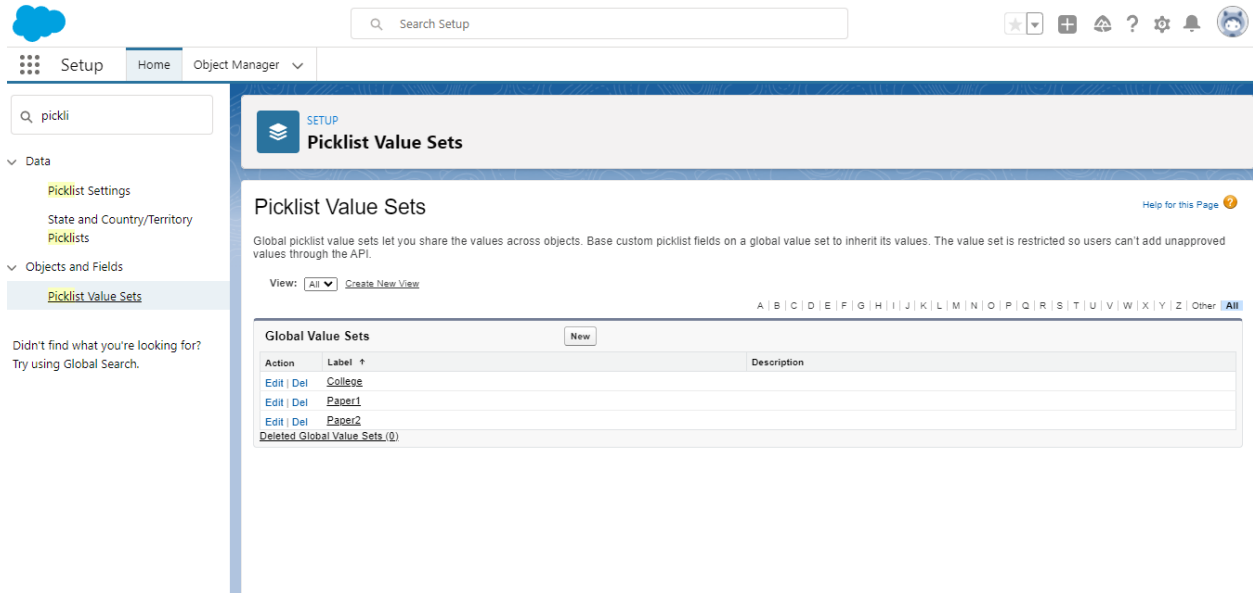


The screenshot shows the Salesforce Setup interface for the 'Subjects' object. The 'Fields & Relationships' section is active, displaying a list of 8 fields. The left sidebar contains navigation links: Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, and Search Layouts. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The top right corner has a search bar and various utility icons.

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		
Students	Students__c	Lookup(Students)		✓

# Task 3 : Adding Business Logic To Application

## Creating Global Picklist Value Sets

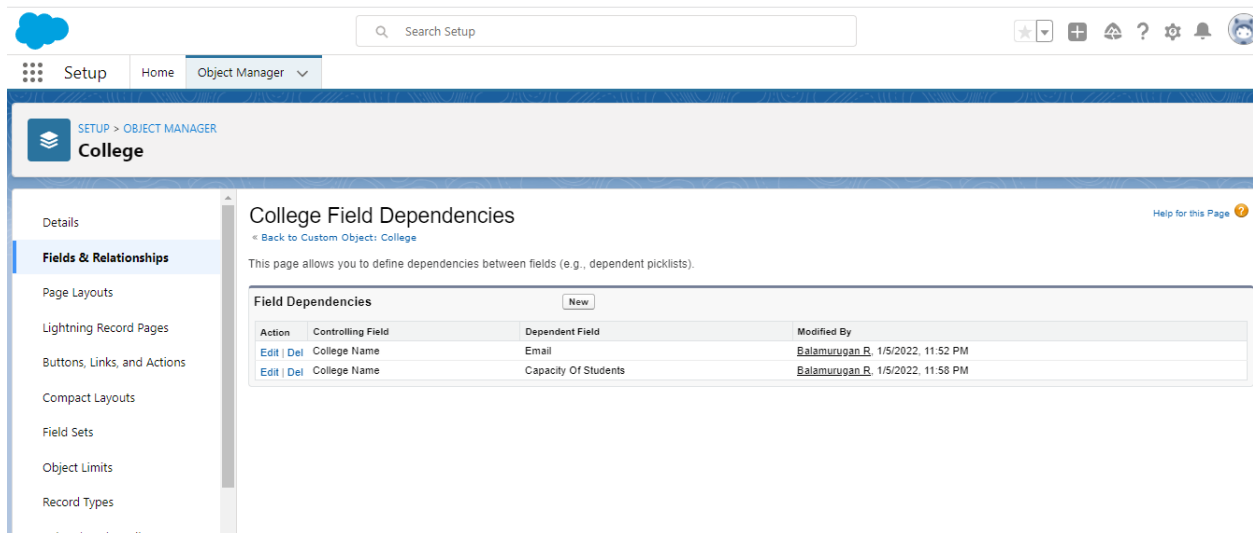


The screenshot shows the Salesforce Setup interface. The left sidebar contains a search bar with 'pickli' and a navigation menu with 'Data' (Picklist Settings, State and Country/Territory, Picklists) and 'Objects and Fields' (Picklist Value Sets). The main content area is titled 'Picklist Value Sets' and includes a description: 'Global picklist value sets let you share the values across objects. Base custom picklist fields on a global value set to inherit its values. The value set is restricted so users can't add unapproved values through the API.' Below the description is a table of 'Global Value Sets' with columns for Action, Label, and Description. The table lists three entries: 'College', 'Paper1', and 'Paper2'. A 'New' button is visible above the table.

Action	Label	Description
<a href="#">Edit</a>   <a href="#">Del</a>	College	
<a href="#">Edit</a>   <a href="#">Del</a>	Paper1	
<a href="#">Edit</a>   <a href="#">Del</a>	Paper2	

Deleted Global Value Sets (0)

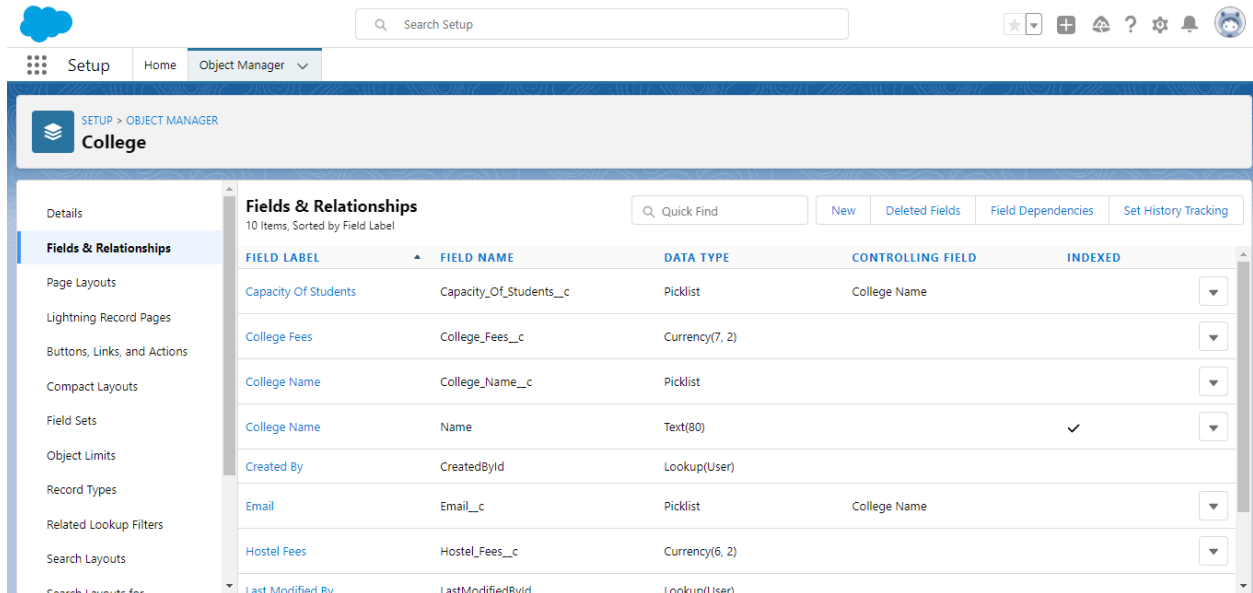
## Creating Field Dependencies



The screenshot shows the Salesforce Setup interface for 'College'. The left sidebar contains a search bar with 'Search Setup' and a navigation menu with 'Details', 'Fields & Relationships', 'Page Layouts', 'Lightning Record Pages', 'Buttons, Links, and Actions', 'Compact Layouts', 'Field Sets', 'Object Limits', 'Record Types', and 'Related Lookup Filters'. The main content area is titled 'College Field Dependencies' and includes a description: 'This page allows you to define dependencies between fields (e.g., dependent picklists).' Below the description is a table of 'Field Dependencies' with columns for Action, Controlling Field, Dependent Field, and Modified By. The table lists two entries: 'College Name' (controlling) and 'Email' (dependent), and 'College Name' (controlling) and 'Capacity Of Students' (dependent).

Action	Controlling Field	Dependent Field	Modified By
<a href="#">Edit</a>   <a href="#">Del</a>	College Name	Email	Balamurugan.R, 1/5/2022, 11:52 PM
<a href="#">Edit</a>   <a href="#">Del</a>	College Name	Capacity Of Students	Balamurugan.R, 1/5/2022, 11:58 PM

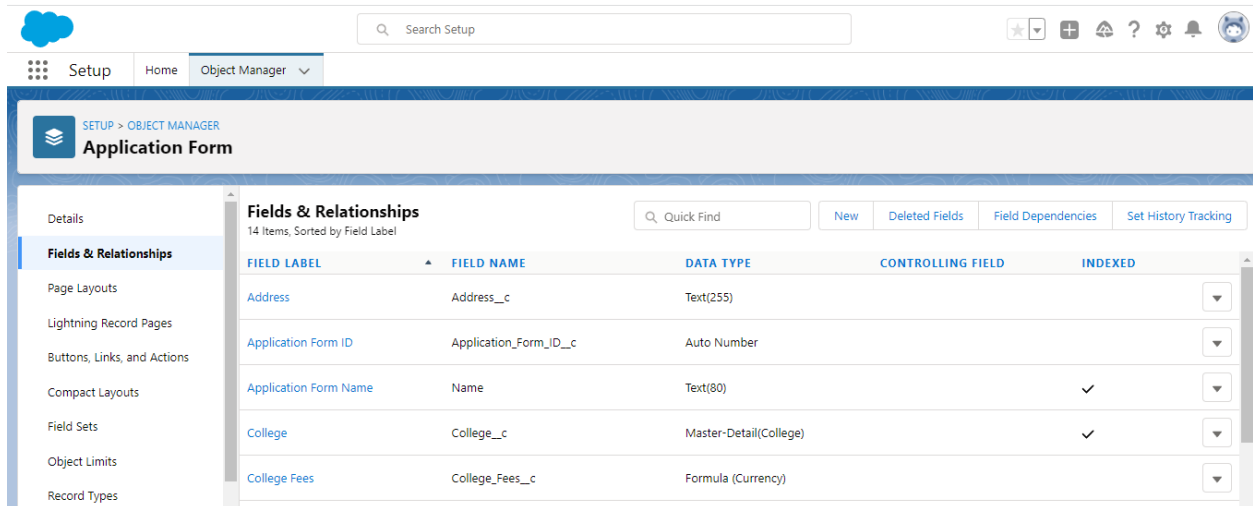
## Validation Rule On College Object



The screenshot shows the Salesforce Setup interface for the 'College' object. The 'Fields & Relationships' tab is selected, displaying a list of 10 fields. The fields are sorted by Field Label. The table includes columns for Field Label, Field Name, Data Type, Controlling Field, and Indexed status.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Capacity Of Students	Capacity_Of_Students__c	Picklist	College Name	
College Fees	College_Fees__c	Currency(7, 2)		
College Name	College_Name__c	Picklist		
College Name	Name	Text(80)		✓
Created By	CreatedById	Lookup(User)		
Email	Email__c	Picklist	College Name	
Hostel Fees	Hostel_Fees__c	Currency(6, 2)		
Last Modified By	LastModifiedById	Lookup(User)		

## Validation Rule on Application Object



The screenshot shows the Salesforce Setup interface for the 'Application Form' object. The 'Fields & Relationships' tab is selected, displaying a list of 14 fields. The fields are sorted by Field Label. The table includes columns for Field Label, Field Name, Data Type, Controlling Field, and Indexed status.

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)		✓
College Fees	College_Fees__c	Formula (Currency)		

# Process Automation

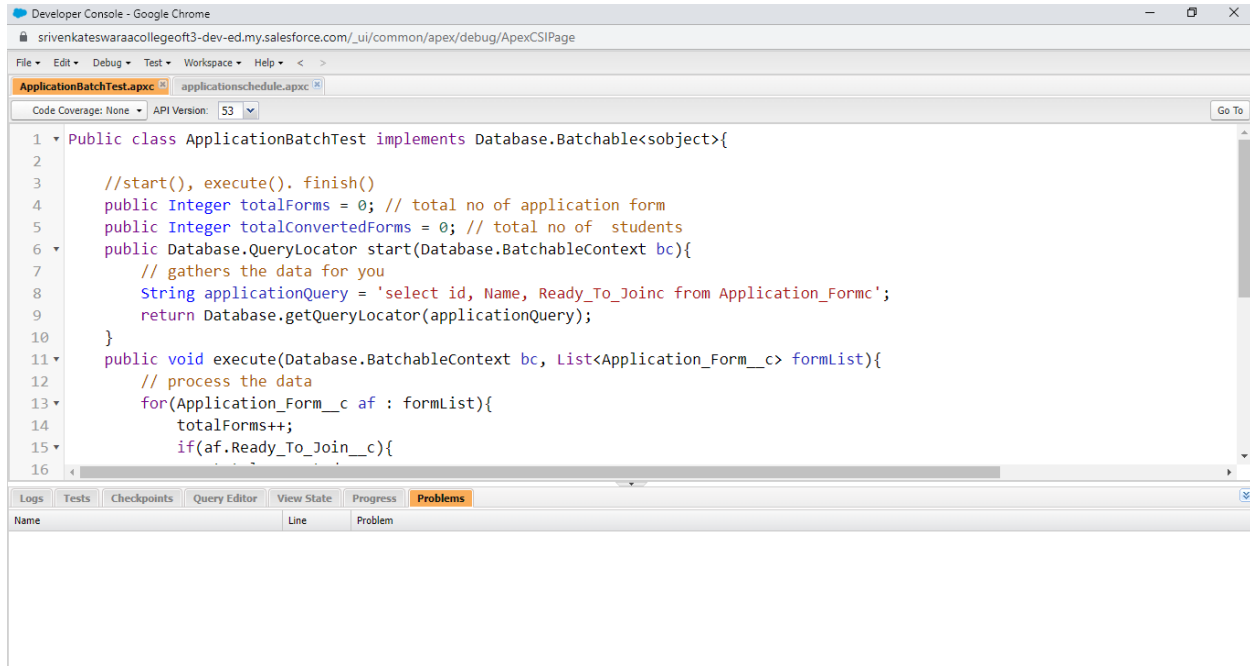
The screenshot displays the Salesforce Process Builder interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The main header is 'Process Builder - Student Record'. The flowchart on the left starts with 'START', followed by an 'Application Form' step, then a decision diamond 'Ready to Join is checked'. If TRUE, it leads to 'IMMEDIATE ACTIONS' containing 'Create a student rec...' (highlighted), 'Email', and 'show fewer'. If FALSE, it leads to another decision diamond 'Add Criteria'. The right panel, titled 'Create a Record', shows the configuration for the 'Create a student record' action. It includes fields for 'Address', 'Date Of Birth', 'Guardian Name', 'Phone', and 'Students Name', all set to 'Field Reference' with values from 'Application\_Form\_...'. The 'Record Type' is set to 'Students'.

## The Student Record Using Flow

The screenshot shows the Salesforce Flow Builder interface for the 'Application Form Trigger Flow - V2'. The left sidebar shows the 'Toolbox' with 'Manager', 'RESOURCES', and 'ELEMENTS'. The main area displays the 'Edit Assignment' configuration for the 'Assign Student Record Data' action. The configuration is titled 'Set Variable Values' and lists several variables to be assigned, each with an 'Operator' (Equals) and a 'Value' (e.g., '\$Record > Address'). The variables include 'StudentVar > Address', 'StudentVar > Application Form', 'StudentVar > College Name', 'StudentVar > Date Of Birth', 'StudentVar > Guardian Name', 'StudentVar > Phone', and 'StudentVar > Student Name'. The 'Value' column shows the corresponding record fields, such as '\$Record > Address', '\$Record > Record ID', '\$Record > College', '\$Record > date of Birth', '\$Record > Guardian Name', '\$Record > Phone', and '\$Record > Student Name'.

## Task 5 : Batch Apex

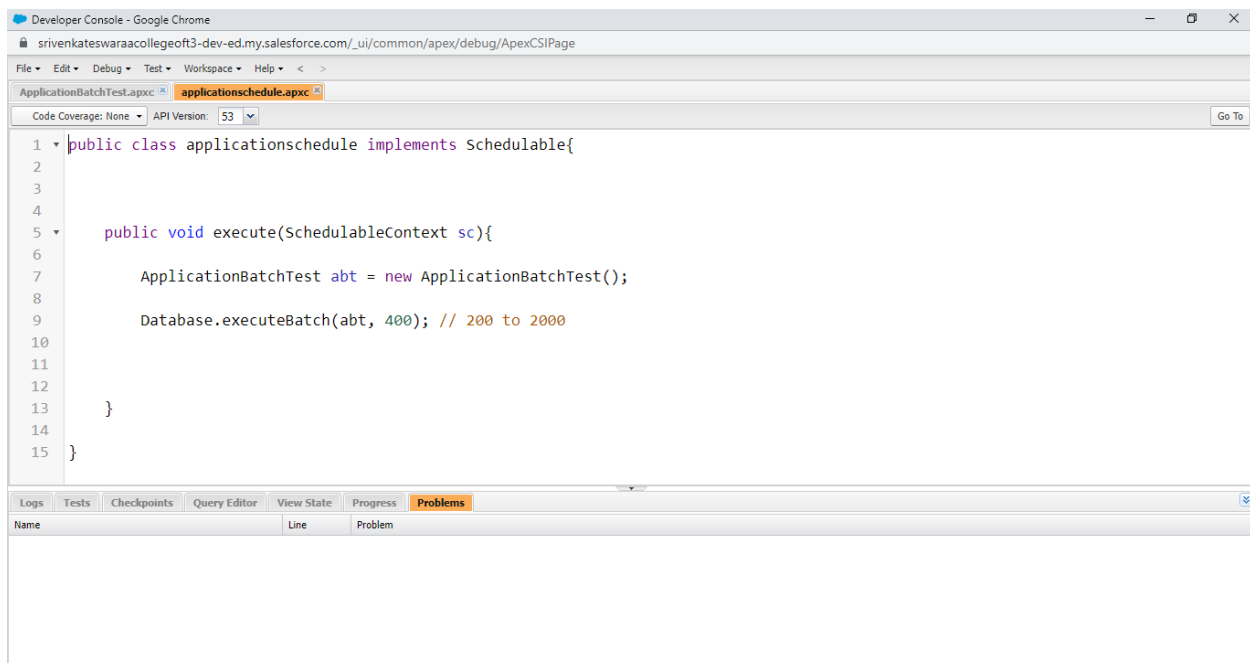
### A Batchapex For Application Form



The screenshot shows the Salesforce Developer Console with the 'ApplicationBatchTest.apxc' file open. The code defines a class that implements the Database.Batchable<Object> interface. It includes methods for start, execute, and finish, along with a query to retrieve application form data. The execute method iterates through a list of application forms and updates a counter.

```
1 Public class ApplicationBatchTest implements Database.Batchable<Object>{
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_Joinc from Application_Formc';
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
```

### A Scheduler Class

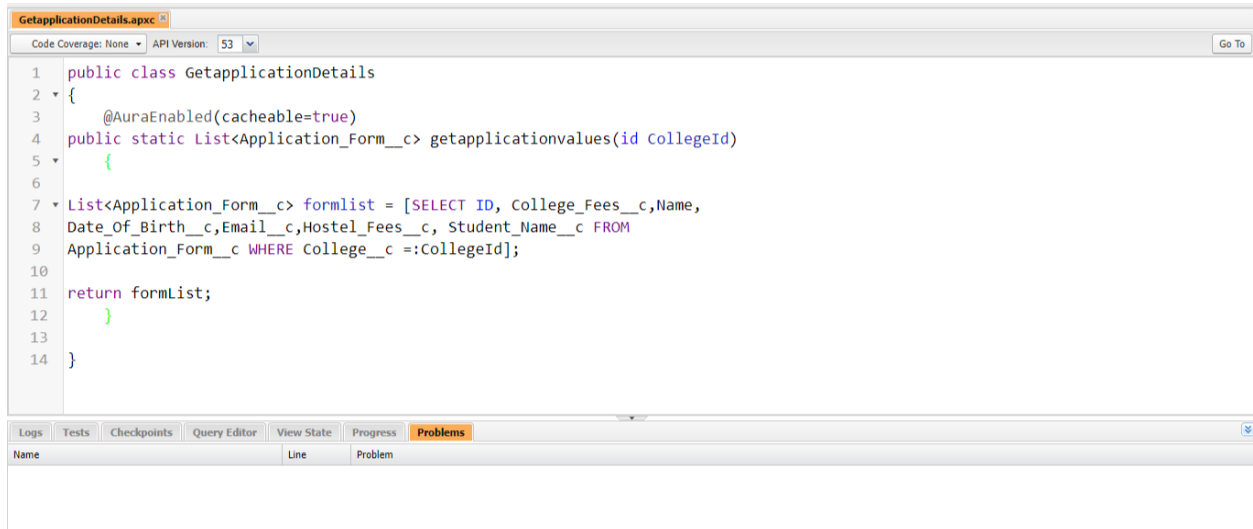


The screenshot shows the Salesforce Developer Console with the 'applicationschedule.apxc' file open. The code defines a class that implements the Schedulable interface. It includes an execute method that creates an instance of ApplicationBatchTest and executes a batch of records.

```
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 }
```

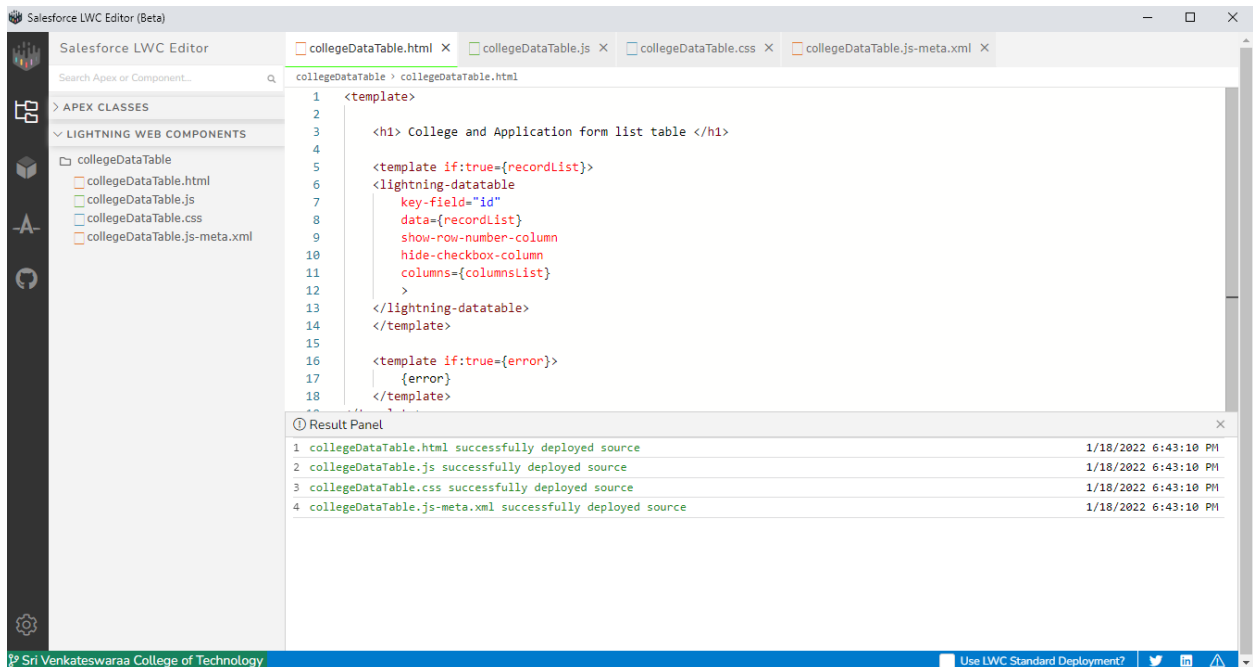
# Task 6 : Lightning Web Components

## College DataTable Component( APEX CLASS)



```
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     }
13
14 }
```

## College DataTable Component (HTML FILE)



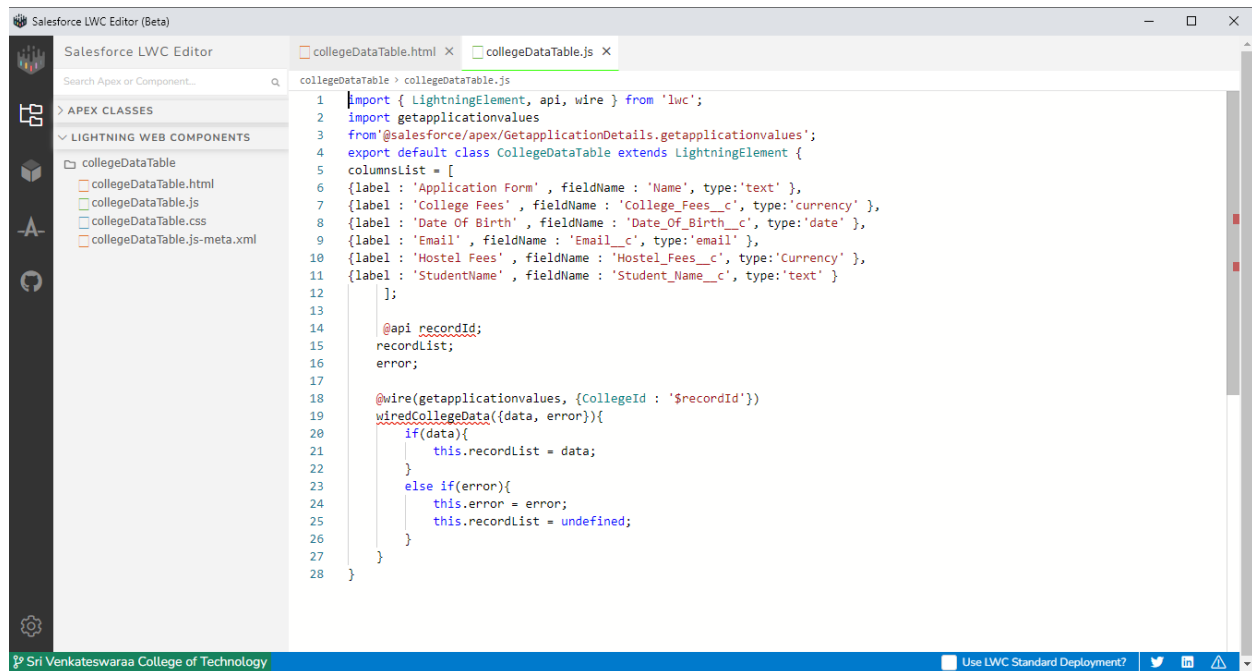
```
1 <template>
2
3     <h1> College and Application form list table </h1>
4
5     <template if:true={recordList}>
6         <lightning-datatable
7             key-field="id"
8             data={recordList}
9             show-row-number-column
10            hide-checkbox-column
11            columns={columnsList}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

Result Panel

Message	Time
1 collegeDataTable.html successfully deployed source	1/18/2022 6:43:10 PM
2 collegeDataTable.js successfully deployed source	1/18/2022 6:43:10 PM
3 collegeDataTable.css successfully deployed source	1/18/2022 6:43:10 PM
4 collegeDataTable.js-meta.xml successfully deployed source	1/18/2022 6:43:10 PM



## College DataTable Component(JAVA SCRIPT FILE)

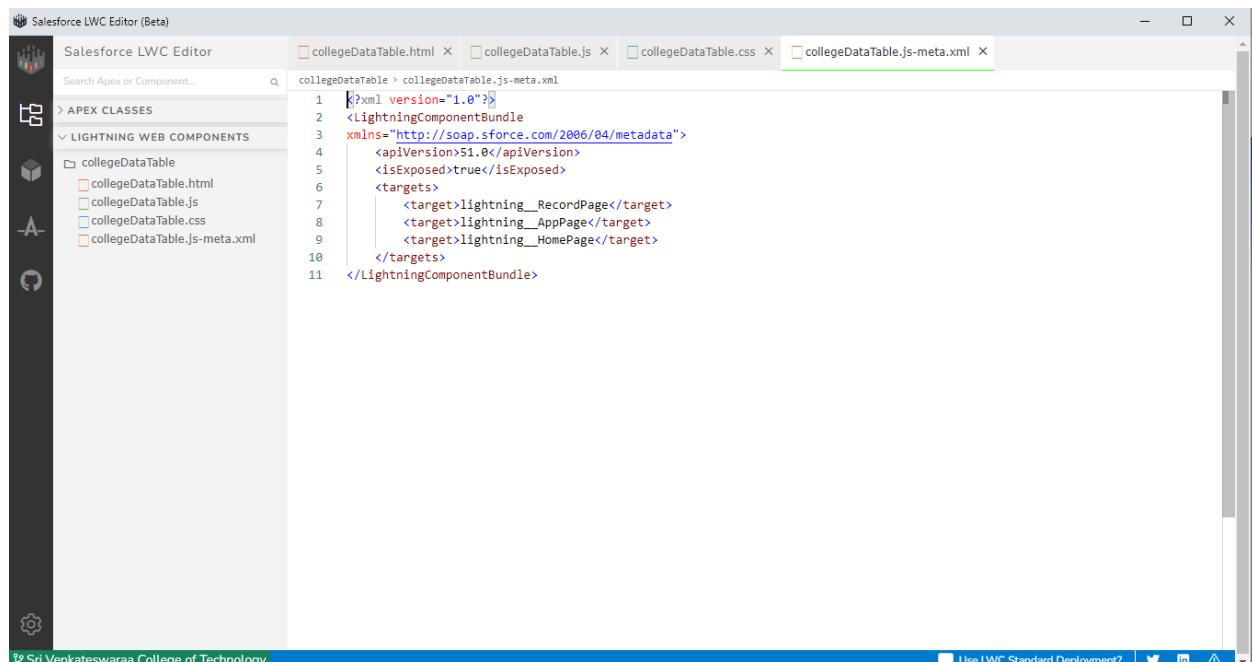


The screenshot shows the Salesforce LWC Editor (Beta) interface. The left sidebar displays the project structure for 'collegeDataTable', including 'collegeDataTable.html', 'collegeDataTable.js', 'collegeDataTable.css', and 'collegeDataTable.js-meta.xml'. The main editor area shows the 'collegeDataTable.js' file with the following code:

```
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;
26    }
27  }
28 }
```

The footer of the editor shows 'Sri Venkateswaraa College of Technology' and a checkbox for 'Use LWC Standard Deployment?'.

## College DataTable Component(META FILE)



The screenshot shows the Salesforce LWC Editor (Beta) interface with the 'collegeDataTable.js-meta.xml' file open. The left sidebar shows the project structure. The main editor area displays the following XML code:

```
1 <?xml version="1.0"?>
2 <LightningComponentBundle
3   xmlns="http://soap.sforce.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 footer of the editor shows 'Sri Venkateswaraa College of Technology' and a checkbox for 'Use LWC Standard Deployment?'.