

College Management Application

Day 1:

Topic : **College Management Application.**

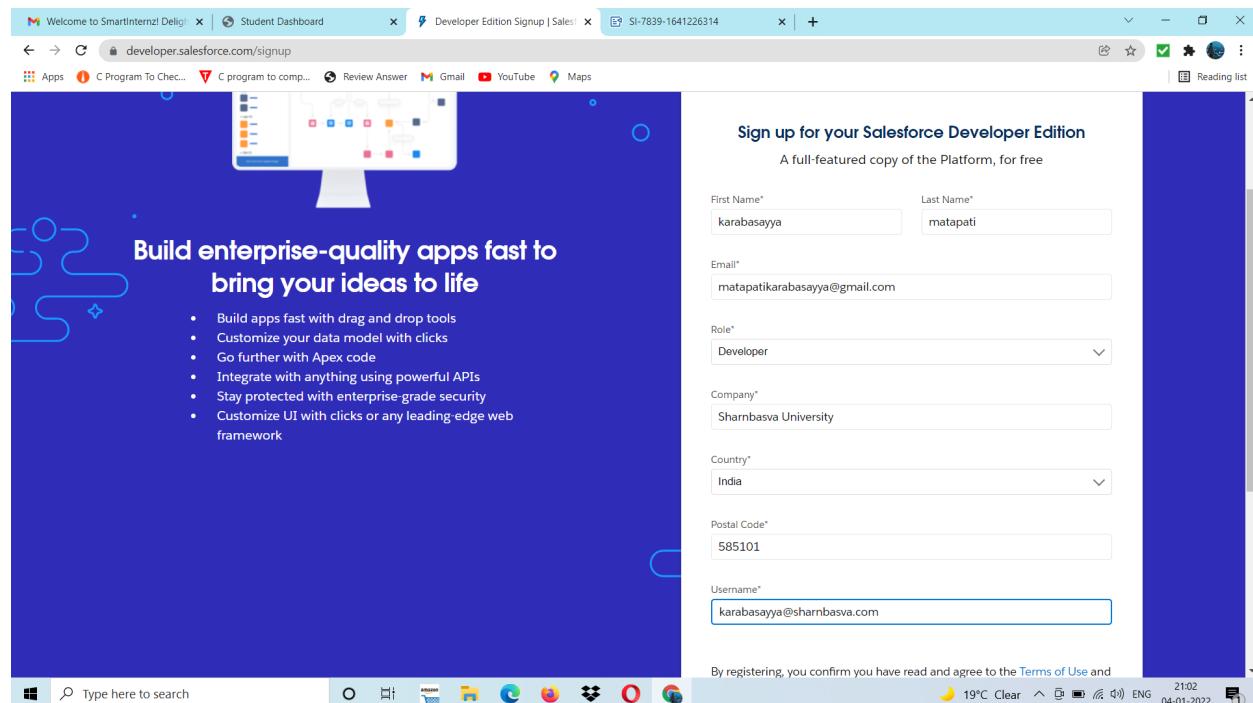
- Milestone/Activities :
- 1)Creating Developer Account.
 - 2)Account Activation.
 - 3) Login To your salesforce account.

Detailed Description : in first step I have created the salesforce developer organization by

following the instructions such as first-name, last-name , e-mail id and username, after that I verified my developer account using e-mail this step activates my salesforce developer account. where I set new password for my developer account and after this I logged into my salesforce developer account using my Username & Password.

screenshot

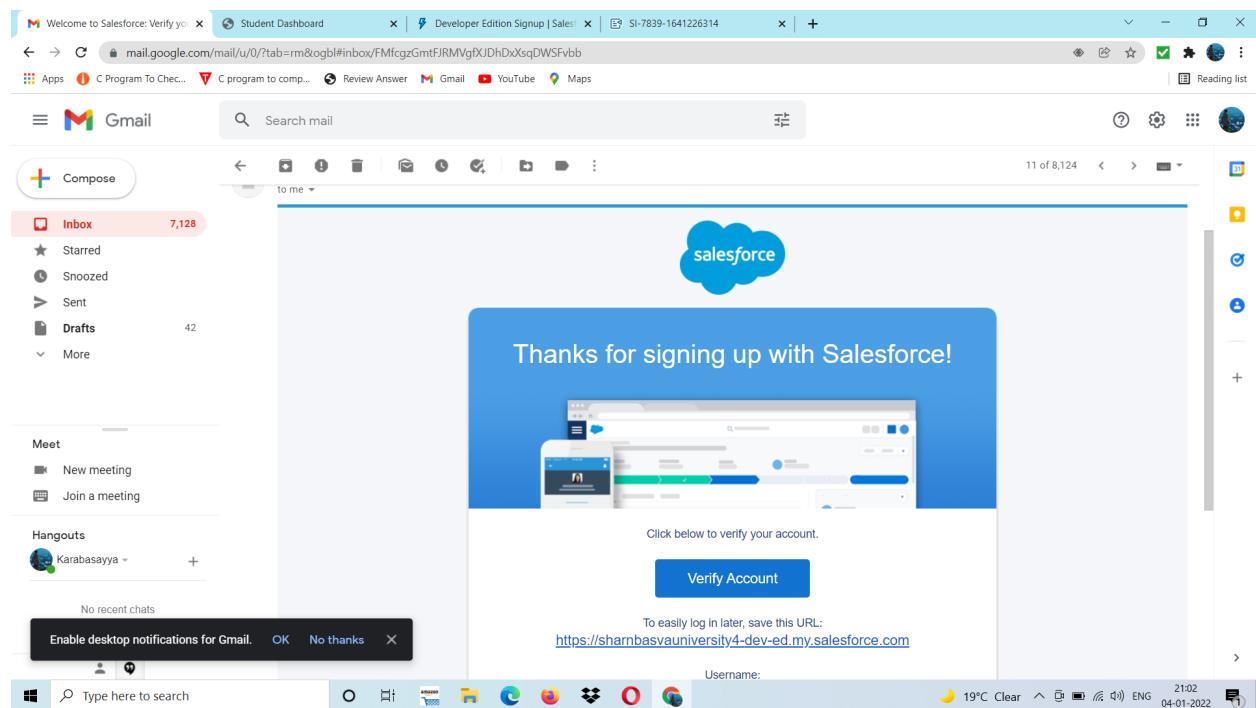
:Creating Developer org Account in [developers.salesforce.com/](https://developer.salesforce.com/signup)



College Management Application

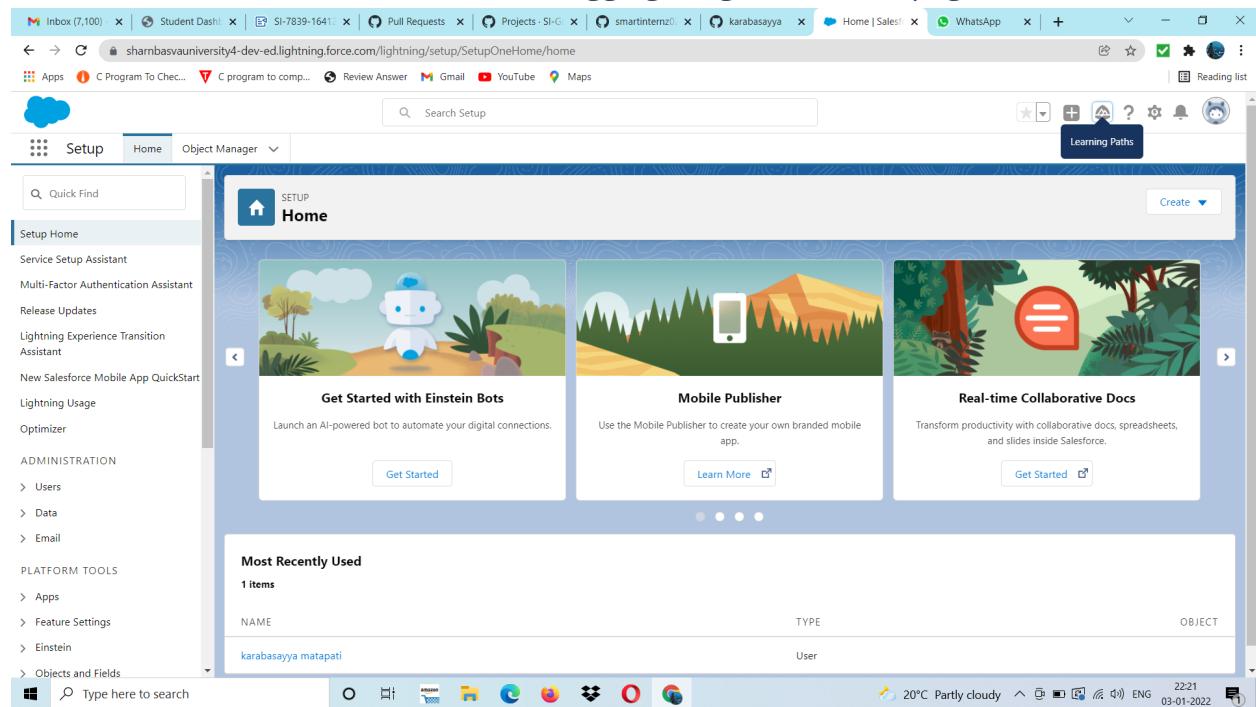
Screenshot

: Account Verification.



Screenshot

: after successful logging in I got this homepage



College Management Application

Day 2 :

Topic : College Management Application.

Milestone/Activities : Create the college management app.

- a) Custom objects creation for the Application
 - 1. College
 - 2. Application Form.
 - 3. Students.
 - 4. Subjects.
- b) Create Fields on College Object.
- c) Create Fields on Application object form

Detailed description: I created the college management app in my salesforce developer org using lightning app, in that app I created the four custom objects in the Object Manager they are,

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

==> Steps for creating the object

a.1)Login-->Setup-->Object Manager-->Create-->Custom Objects.

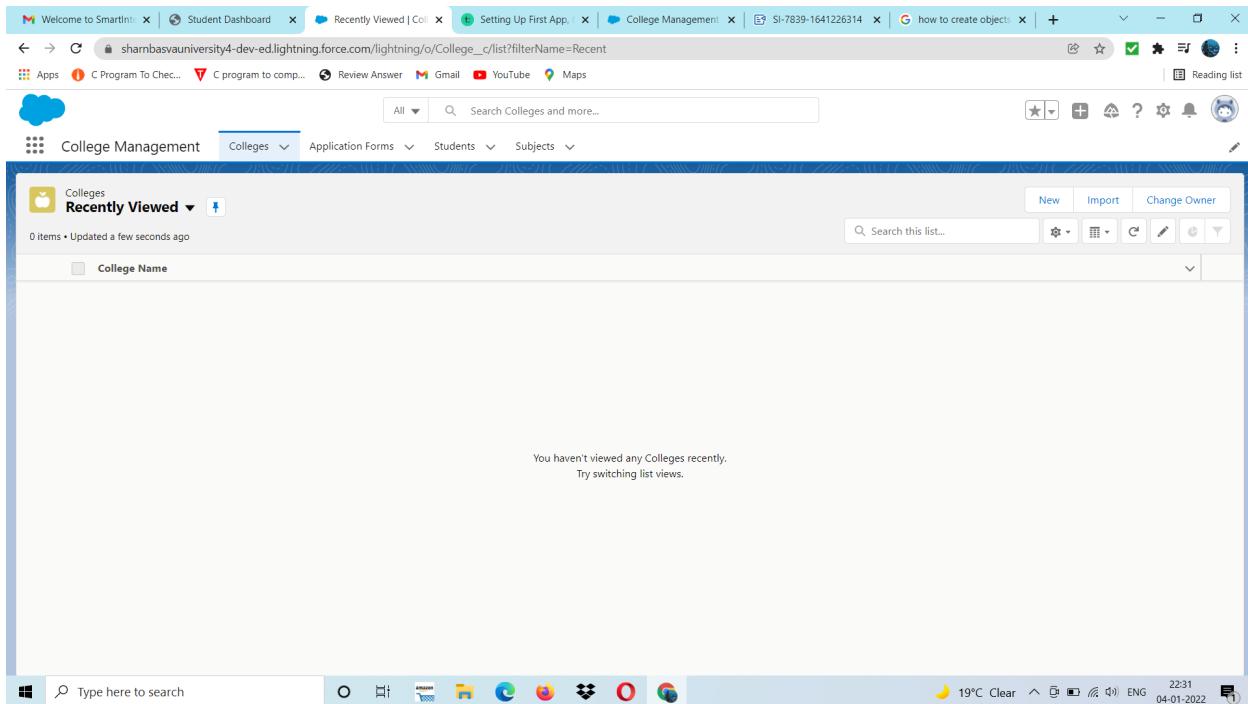
after creation of the objects I created the fields for the each object in the Fields and Relationships

==> steps for creating the Fields in Objects

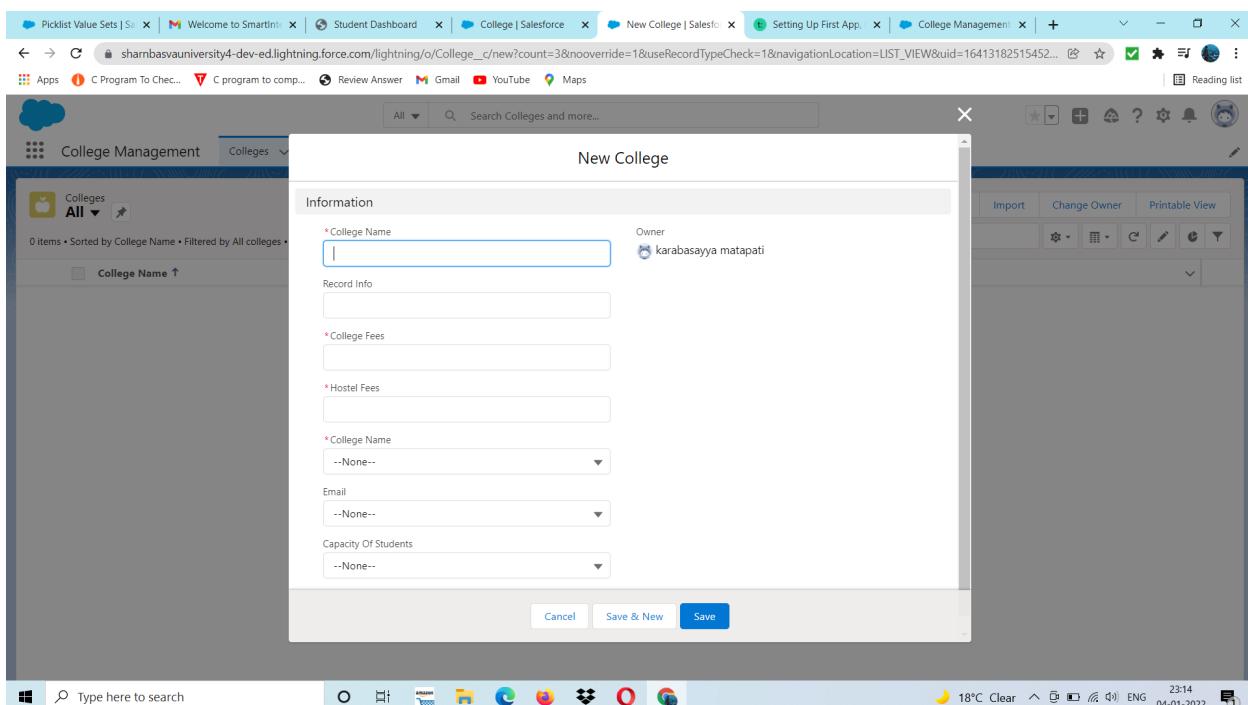
b.1)Login-->Setup-->Object Manger-->Fields & Relationship-->New.

College Management Application

Screenshot : college management app and its objects

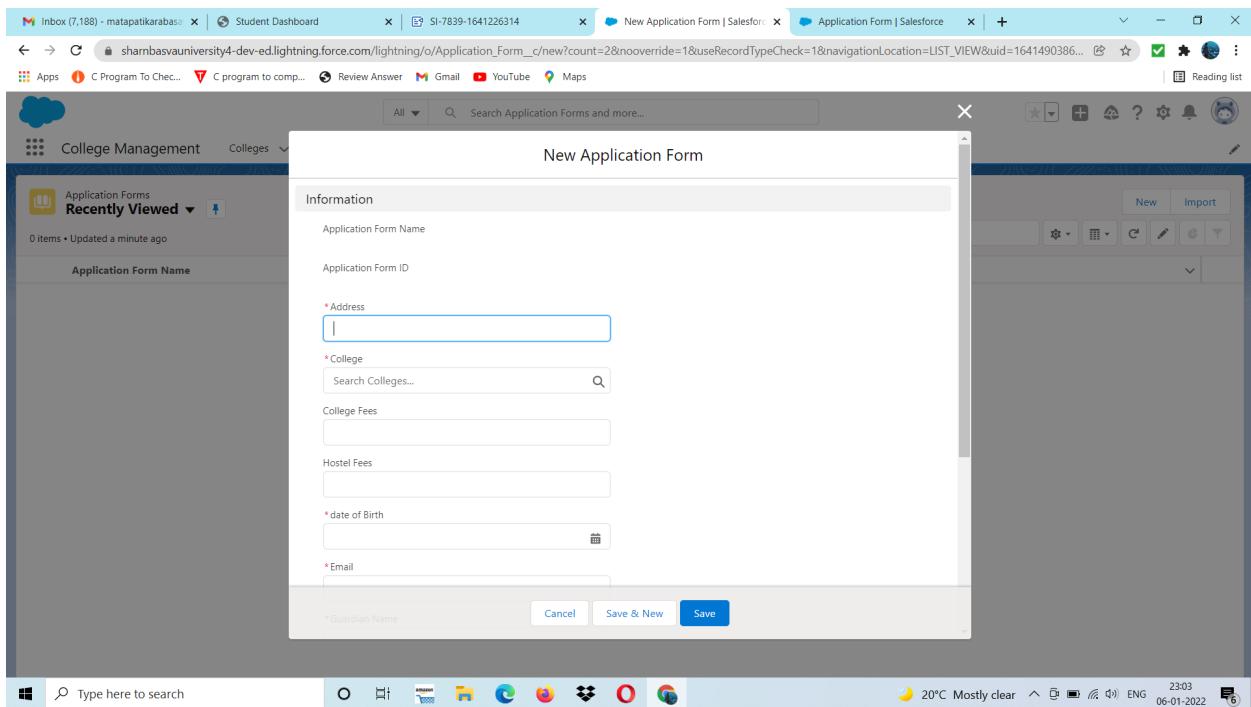


Screenshot : Created the Fields for College object



College Management Application

Screenshot: Created the Fields on Application Form Object



College Management Application

Day3 :

Topic : **College Management Application.**

- Milestone/Activities :
- 1)Created The Fields on Student Object.
 - 2)Created The Fields on Subject Object.
 - 3)Creating Global Picklist Value Sets

Detailed description: I logged into the salesforce developer org where I created the fields for the Student object , and also created the Fields for Subject objects , after this I created the Global pick List Value Sets.

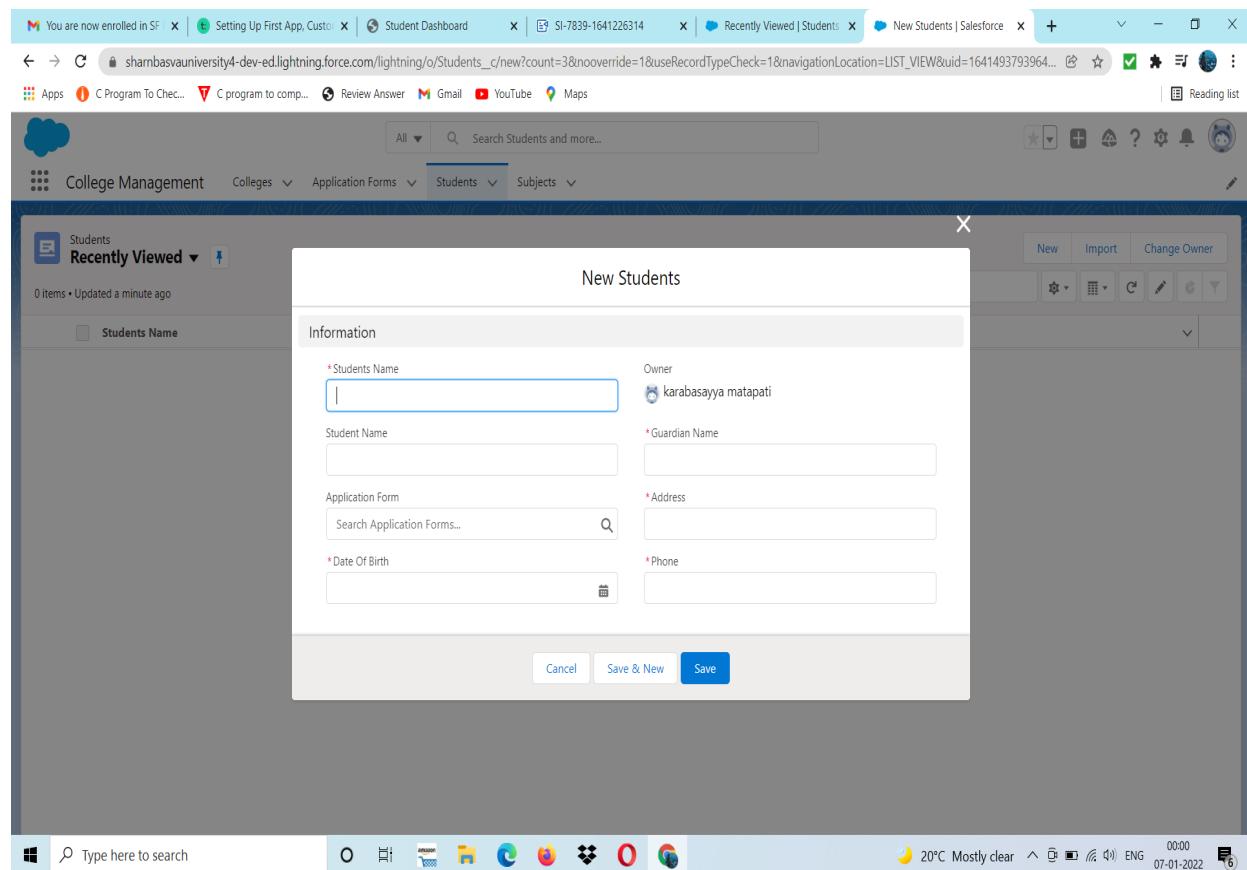
==>Steps for creating The fields on object

a)Login-->Setup-->Object Manager-->object-->new--> save.

==>Steps for creating PickList Value Sets

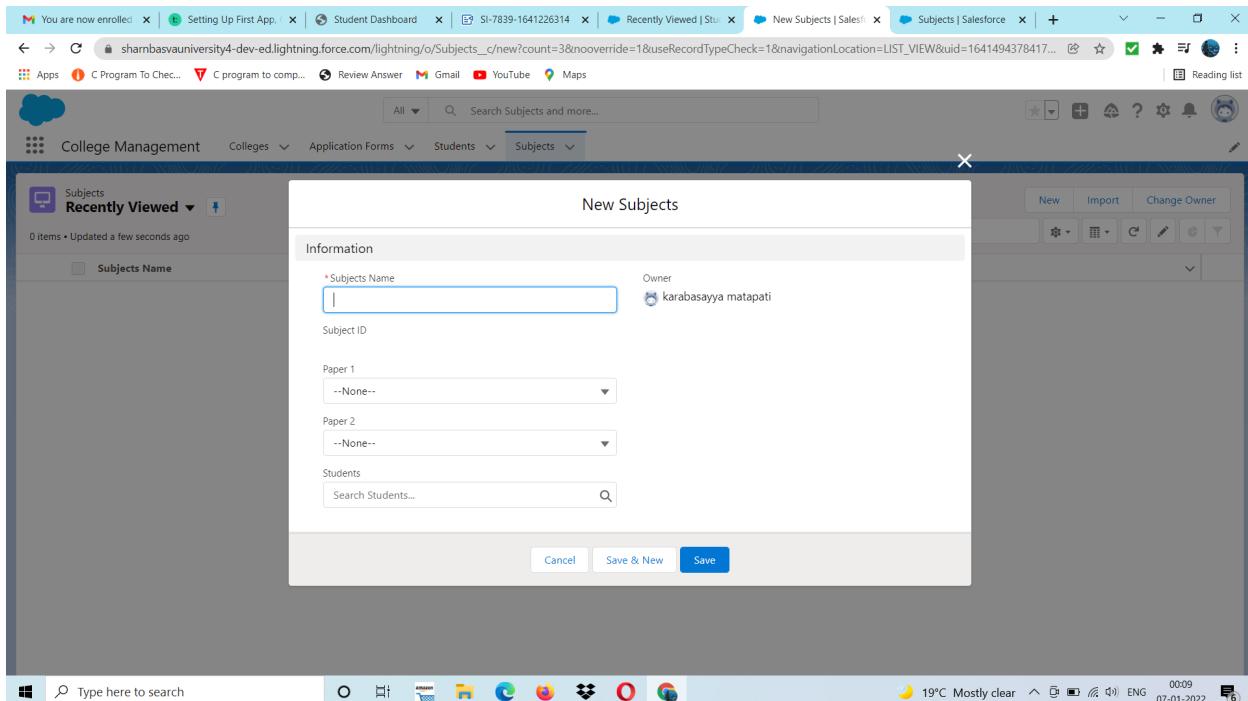
b)Login-->Setup-->Home-->PickList Value Sets-->new-->Save.

Screenshot : Created the Fields on Student Object.

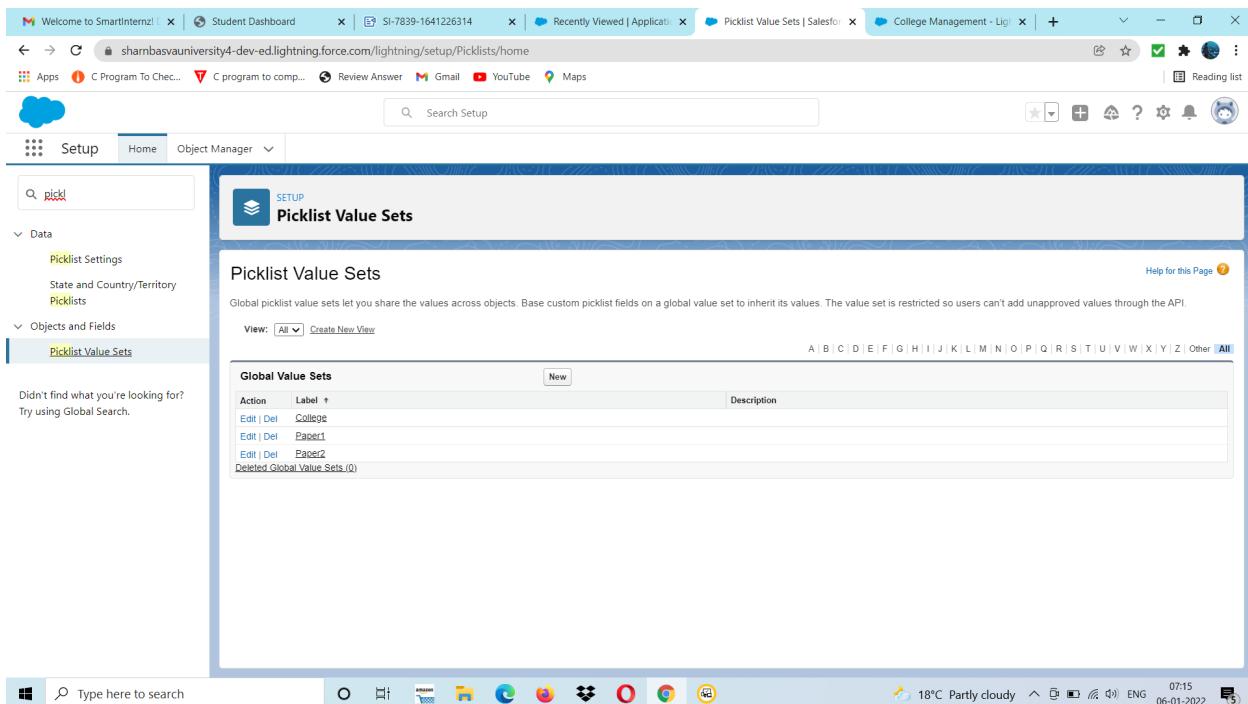


College Management Application

Screenshot : Created The Fields on Subjects Object



Screenshot: Created the Global PickList Value Sets



College Management Application

Day 4:

Topic : **College Management Application.**

Milestone/Activities : 1)Creating Field Dependencies

2)Creating Validation Rules

Detailed description: 1) created the field dependencies for college object where, where the controlling field is college Name and dependent field is Email. while creating the field dependencies , Selected the email ids according to the college name. after this I created the one more field dependencies that is between college Name and capacity of students, where the controlling field is college Name and dependent field is Capacity of Students.

- 2) a)created the validation rules for college object and application in first validation rule the college name and record info need to same if you enter the wrong name it displays the error message
- b) created the validation rule for the applicatin form object where once you created the record further you cannot modify it.

College Management Application

Screenshot : Creating Field Dependencies

1) College name <==>Email

The screenshot shows the Salesforce Setup interface for creating field dependencies. The page title is "College Field Dependencies". The left sidebar is titled "Fields & Relationships" and includes options like Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, and Search Layouts for Salesforce Classic. The main content area displays a table titled "Field Dependencies" with one row:

Action	Controlling Field	Dependent Field	Modified By
Edit Del	College Name	Email	karabasavya.mataoati 1/7/2022, 10:14 PM

2)College name<==>Capacity of Students

The screenshot shows the Salesforce Setup interface for creating field dependencies. The page title is "College Field Dependencies". The left sidebar is titled "Fields & Relationships" and includes options like Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, and Search Layouts for Salesforce Classic. 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	karabasavya.mataoati 1/7/2022, 10:14 PM
Edit Del	College Name	Capacity Of Students	karabasavya.mataoati 1/7/2022, 10:22 PM

College Management Application

Screenshot: created the validation rules.

a) validation rule for College object

The screenshot shows the Salesforce Setup interface for the 'College' object. On the left, a sidebar lists various configuration options like Details, Fields & Relationships, Page Layouts, etc. The main area is titled 'Validation Rules' and shows one item: 'College_name_similar'. The table has columns for Rule Name, Error Location, Error Message, Active, and Modified By. The rule 'College_name_similar' is active and was modified by 'karabasayya matapati' on 1/7/2022 at 10:58 PM. The error message is 'Please use the college name in record info'. The error location is 'Top of Page'.

b) Created the validation rule for Application Form

The screenshot shows the Salesforce Setup interface for the 'Application Form' object. The sidebar and validation rules table are identical to the 'College' object screenshot. It shows one validation rule named 'to_stop_modification' with the same details: active, modified by 'karabasayya matapati' on 1/7/2022 at 11:17 PM, error message 'once you create the record further you can't modify it', and error location 'Top of Page'.

College Management Application

Day 5:

Topic : College Management Application.

Milestone/Activities : 1)Process Automation

2)Create The Student Record Using Flow

Detailed description: Created 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, and activated it .
after this I deactivated the process automation and worked on the flows there are mainly 5 types of flows and they are

- 1) Screen Flow
- 2)Record Triggered Flow
- 3)Schedule Triggered Flow
- 4)Platform event triggered Flow
- 5)Auto launched Flow

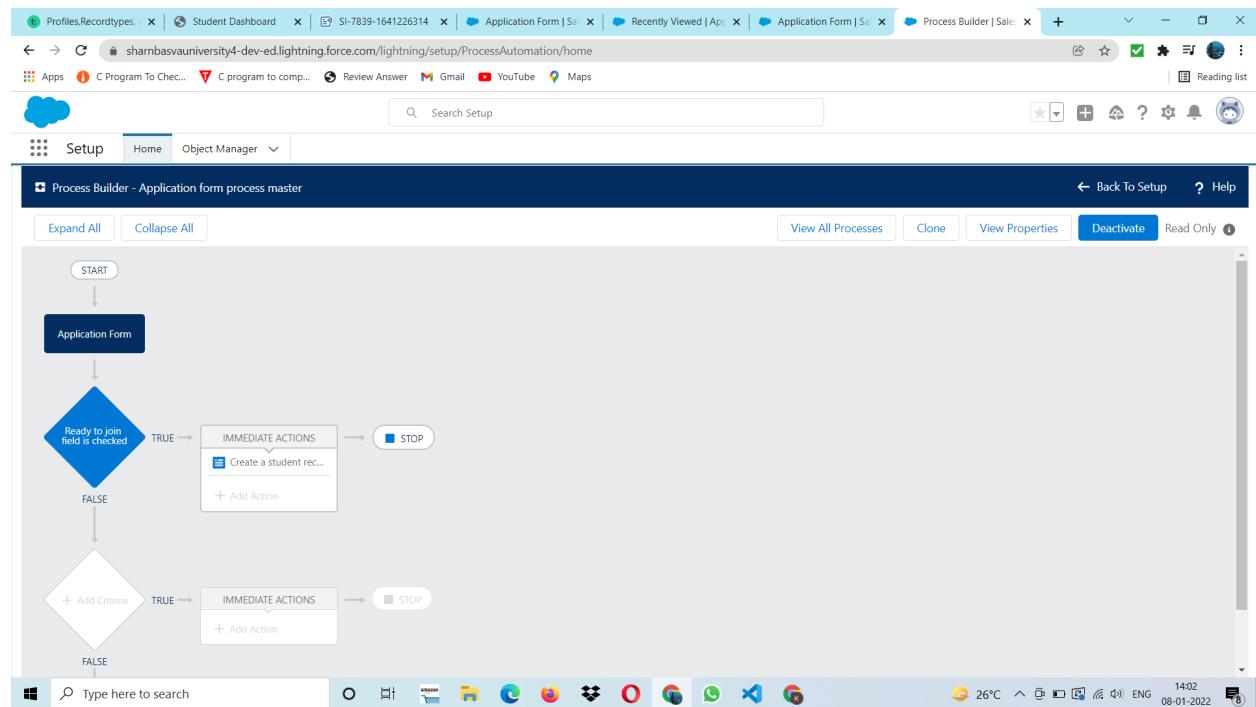
We created the Student Record using Record Triggered Flow.
Creating or updating a record can trigger an auto launched flow to make additional updates to that record before it's saved to the database. A record-triggered flow can update a Salesforce record 10 times faster than a record-change process.

Security model in salesforce

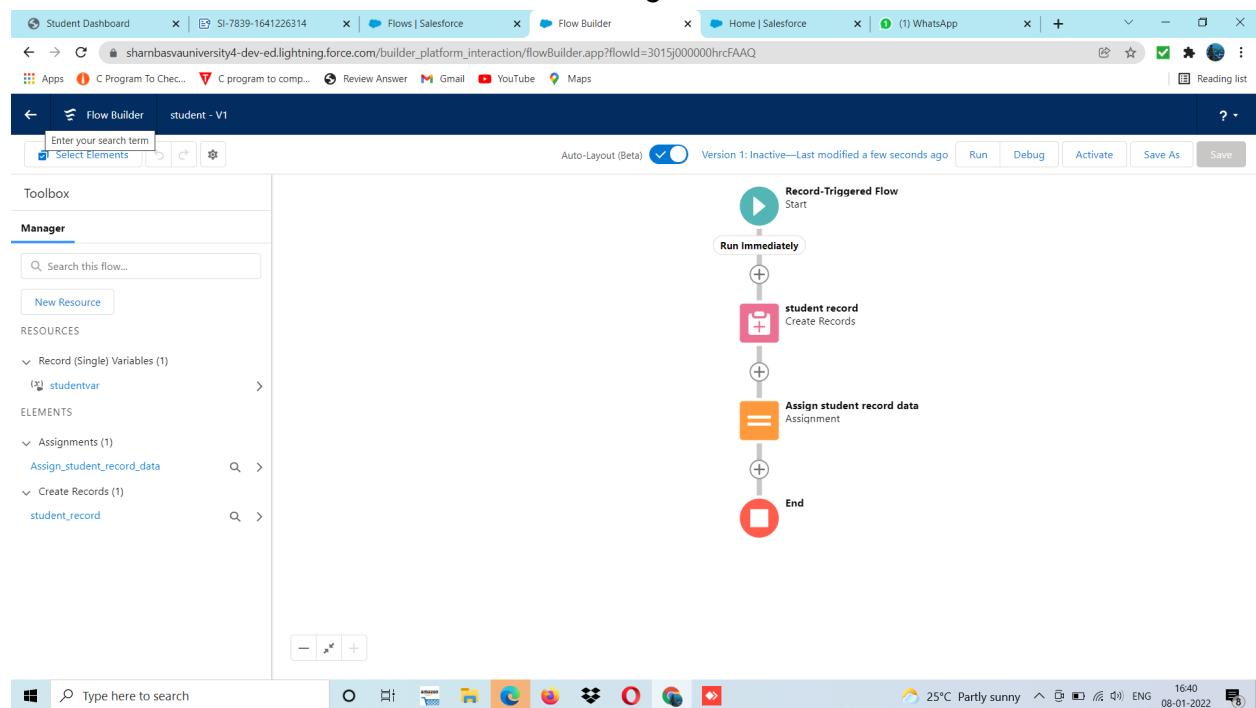
- 1) Object level security
- 2) Field level Security
- 3)Record level Security

College Management Application

Screenshot: Process automation



Screenshot: Created The Student Record Using Flow



College Management Application

Day 6:

Topic : College Management Application.

Milestone/Activities : Create A Batch apex For Application Form

Detailed description: Apex is a strongly typed, object-oriented programming language that allows developers to execute flow and transaction control statements on Salesforce servers in conjunction with calls to the API. Using syntax that looks like Java and acts like database stored procedures, Apex enables developers to add business logic to most system events, including button clicks related record updates, and Visual force pages. Apex code can be initiated by Web service requests and from triggers on objects.

in apex we have primitive datatypes like int,double and boolean, string,date,datetime,id etc..

in apex we have Collections List,Set,Map.

I created the ApplicationBatchTest Class where I executed the code.

The Apex programming language is saved and runs in the cloud—the multitenant platform. Apex is tailored for data access and data manipulation on the platform, and it enables you to add custom business logic to system events. While it provides many benefits for automating business processes on the platform, it is not a general purpose programming language.

College Management Application

ScreenShot: created the apex class named as ApplicationBatchTest

The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Google Chrome" and the URL is "sharnbasvauniversity4-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The main area displays the Apex code for "ApplicationBatchTest.apxc". The code implements the Database.Batchable<sObject> interface. It initializes counters for total forms and converted forms. In the start() method, it queries application form records. In the execute() method, it processes each record to check if it's ready to join. If so, it increments the converted forms counter. In the finish() method, it sends an email to an admin with a summary of the processed data. Below the code editor is a tabs bar with "Logs" selected, followed by "Tests", "Checkpoints", "Query Editor", "View State", "Progress", and "Problems". At the bottom is a toolbar with various icons and a status bar showing "27°C Sunny" and the date "15-01-2022".

```
12 // process the data
13 for(ApplicationForm__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[] emailAddress = new String[]{'matapatikarabasayya@gmail.com'};
27     mail.setToAddresses(emailAddress);
28     Messaging.sendEmail(new Messaging.SingleEmailMessage[]{ mail });
29 }
30 }
```

This screenshot is similar to the first one, showing the Salesforce Developer Console with the "ApplicationBatchTest.apxc" code. However, a modal dialog titled "Enter Apex Code" is overlaid on the bottom right. The dialog contains two lines of code: "1 ApplicationBatchTest abt= new ApplicationBatchTest();" and "2 Database.executeBatch(abt,400);". At the bottom of the developer console, there is a toolbar with "Open Log", "Execute", and "Execute Highlighted" buttons.

```
1 Public class ApplicationBatchTest implements Database.Batchable<sObject>{
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,
9             return Database.getQueryLocator(applicationQuery);
10    }
11    public void execute(Database.BatchableContext bc) {
12        // process the data
13        for(ApplicationForm__c af : formList){
14            totalForms++;
15            if(af.Ready_To_Join__c){
16                totalConvertedForms++;
17            }
18        }
19    }
20 }
```

College Management Application

Day 7 :

Topic : **College Management Application.**

Milestone/Activities : Create A Schedular Class.

Detailed description: To invoke Apex classes to run at specific times, first implement the Schedulable interface for the class, then specify the schedule using either the Schedule Apex page in the Salesforce user interface, or the System.schedule method.

To schedule an Apex class to run at regular intervals, first write an Apex class that implements the Salesforce-provided interface Schedulable. The scheduler runs as system—all classes are executed, whether or not the user has permission to execute the class. To monitor or stop the execution of a scheduled Apex job using the Salesforce user interface, from Setup, enter Scheduled Jobs in the Quick Find box, then select

Scheduled Jobs.

The Schedulable interface contains one method that must be implemented, execute.

```
global void execute(SchedulableContext sc){}
```

The implemented method must be declared as global or public. Use this method to instantiate the class you want to schedule.

College Management Application

Screenshot : Created a Schedular Class

The screenshot shows the Salesforce Developer Console interface. At the top, the URL is `sharmbasavauniversity4-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage`. The tab bar shows `ApplicationBatchTest.apxc` and `ApplicationSchedule.apxc`. Below the tabs, there are dropdowns for `Code Coverage: None` and `API Version: 53`.

```
1 public class applicationschedule implements Schedulable{
2
3
4
5     public void execute(SchedulableContext sc){
6         ApplicationBatchTest abt = new ApplicationBatchTest();
7
8         Database.executeBatch(abt, 400); // 200 to 2000
9
10    }
11
12
13
14
15
16 }
17
18 |
```

Below the code editor is a log table with the following data:

User	Application	Operation	Time	Status	Read	Size
karabasaya matapati	Unknown	Batch Apex	1/15/2022, 12:57:38 PM	sObject type 'ApplicationForm__c' is n...	Unread	4.61 KB
karabasaya matapati	Unknown	/services/data/v53.0/tooling/executeA...	1/15/2022, 12:57:37 PM	Success	Unread	3.36 KB

At the bottom of the interface, there is a search bar with the placeholder "Type here to search" and a toolbar with various icons.

College Management Application

Day 8:

Topic : **College Management Application.**

Milestone/Activities : LWC Component

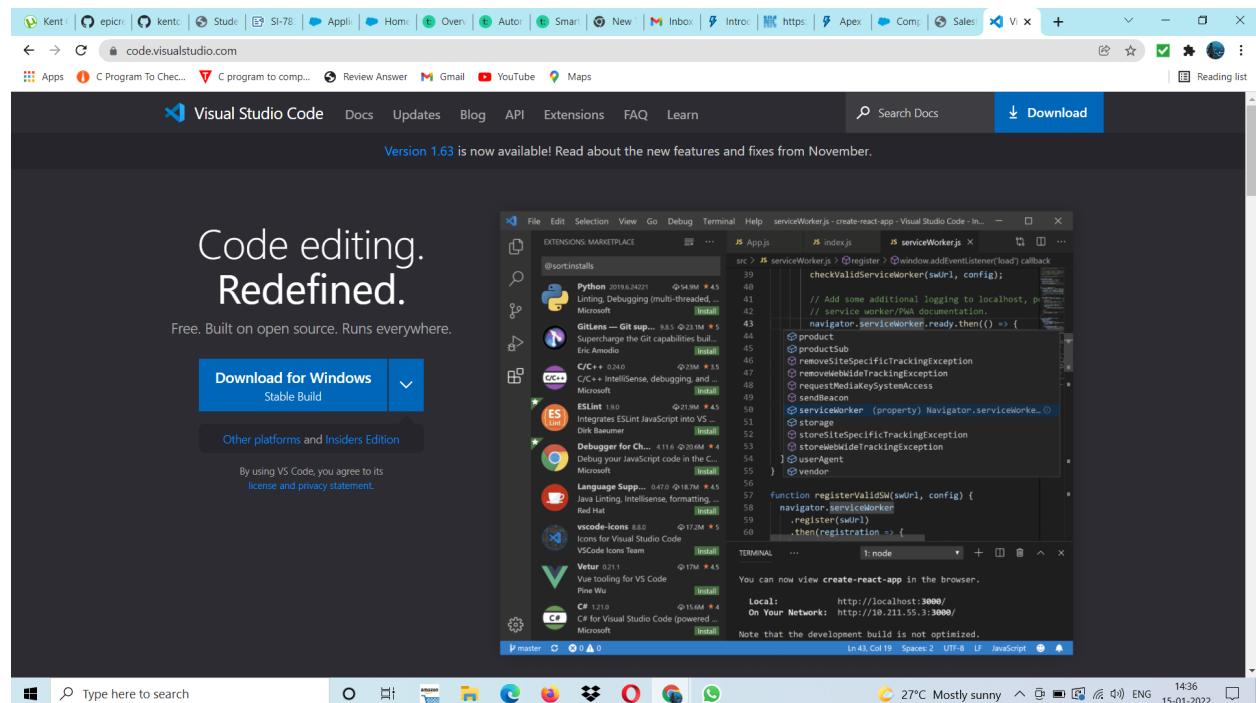
- 1) Installation of visual studio
- 2) installation of salesforce extension package

Detailed Description : Lightning Web Components (LWC) is a programming model for a Lightning Component Framework released in Spring 2019, though available since December 2018. A Lightning web component is a reusable custom HTML element with its own API.

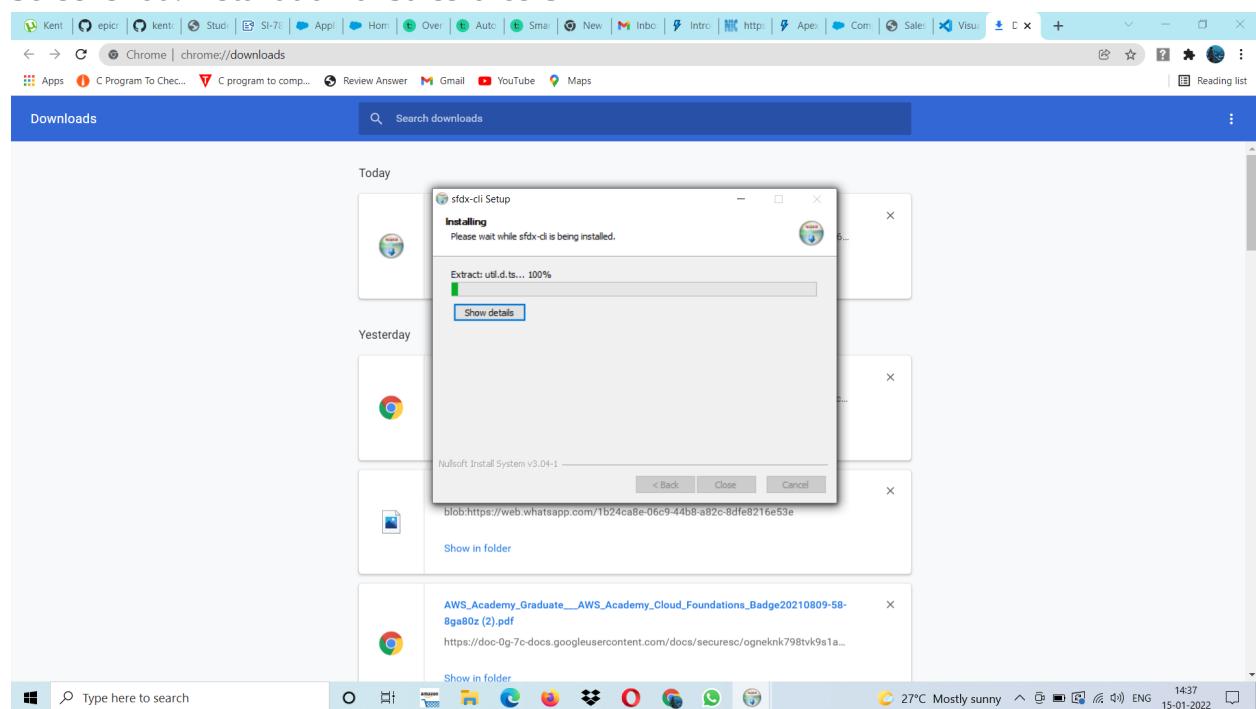
First we need to install the visual studio code and after that we needed to install the salesforce extension pack, and also we need to install the salesforce CLI after this all steps we need to connect our org to the visual studio through the salesforce extension, after that we can access all the files in the visual studio code only.

College Management Application

ScreenShot : installation of Visual Studio

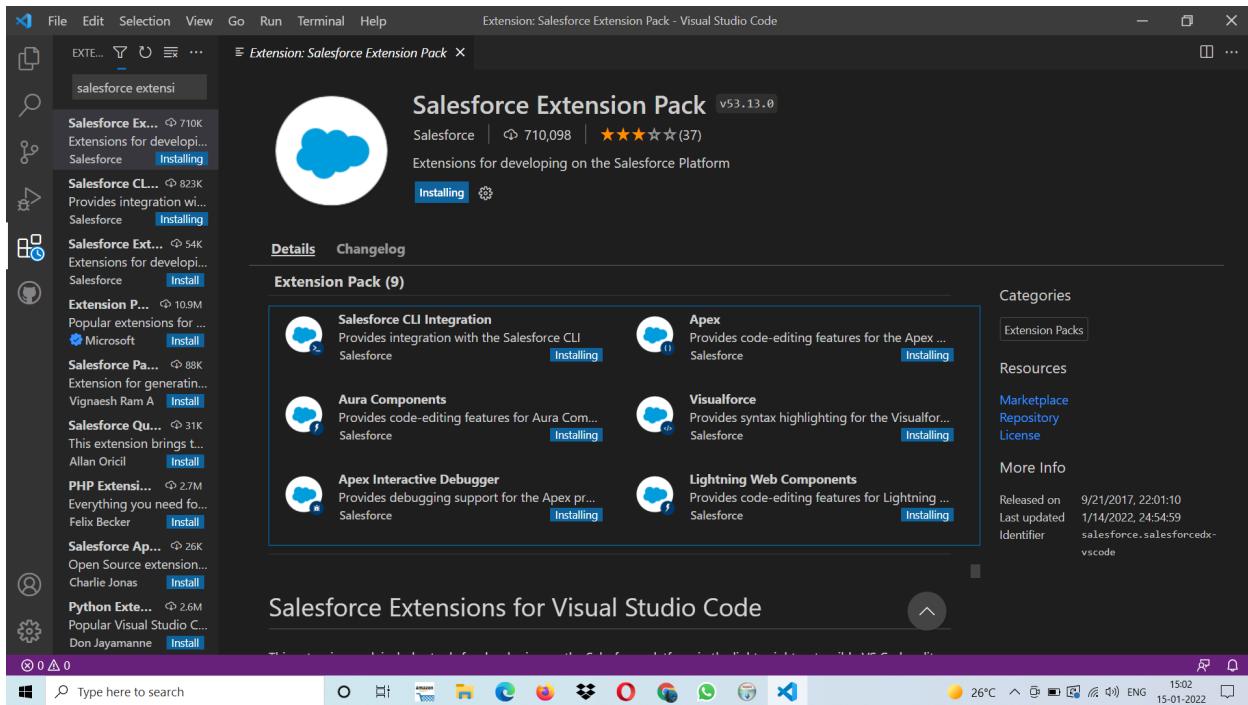


Screenshot : installation of Salesforce CLI



College Management Application

ScreenShot : Installation of salesforce Extension pack



College Management Application

Day 9:

Topic : **College Management Application.**

Milestone/Activities : LWC Components

Introduction to the Javascript

Detailed Description : Now you can build Lightning components using two programming models: Lightning Web Components, and the original model, Aura Components. Lightning web components are custom HTML elements built using HTML and modern JavaScript. Lightning web components and Aura components can coexist and interoperate on a page. To admins and end users, they both appear as Lightning components.

Define a Component:

A Lightning web component that renders UI must include an HTML file, a JavaScript file, and a metadata configuration file. The files must use the same name so the framework can autowire them. A service component (library) must include a JavaScript file and a metadata configuration file.

HTML Templates:

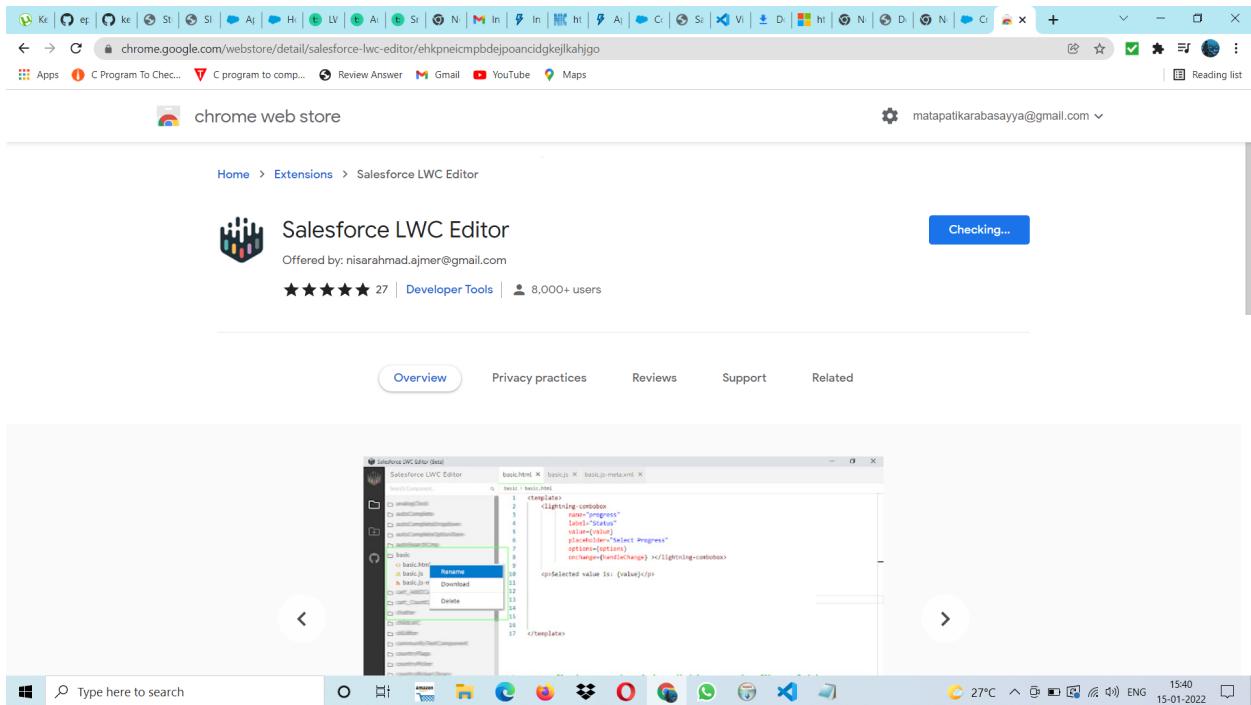
The power of Lightning Web Components is the templating system, which uses the virtual DOM to render components smartly and efficiently. It's a best practice to let LWC manipulate the DOM instead of writing JavaScript to do it.

CSS :

To give your component the Lightning Experience look and feel, use Lightning Design System. To go your own way, write your own CSS.

College Management Application

Screenshot : Adding chrome Extension of salesforce Lwc Editor



College Management Application

Day 10:

Topic : College Management Application.

Milestone/Activities : LWC Component

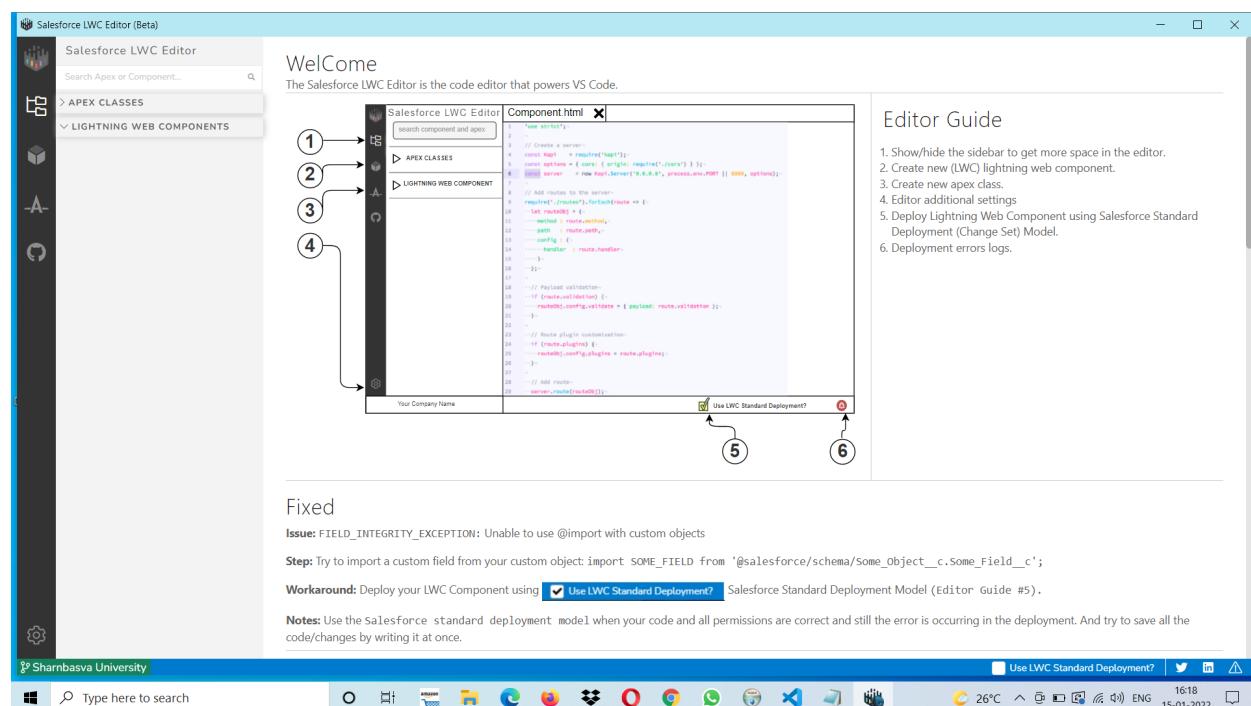
Detailed Description : Using LWC Editor we can deploy the our data into salesforce org, we have learnt about the javaScript datatypes, methods.

JavaScript is a lightweight, cross-platform, and interpreted scripting language. It is well-known for the development of web pages, many non-browser environments also use it. JavaScript can be used for Client-side developments as well as Server-side developments. JavaScript contains a standard library of objects, like Array, Date, and Math, and a core set of language elements like operators, control structures, and statements.

A function is a set of statements that take inputs, do some specific computation, and produces output. Basically, a function is a set of statements that performs some tasks or does some computation and then return the result to the user.

We have worked with the LWC editor

ScreenShot : salesforce LWC EDitor



College Management Application

ScreenShot : salesforce LWC EDitor

```
1 Public class ApplicationBatchTest implements Database.Batchable<sObject>{
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 ApplicationForm__c';
9         return Database.getQueryLocator(applicationQuery);
10    }
11    public void execute(Database.BatchableContext bc, List<ApplicationForm__c> formList){
12        // process the data
13        for(ApplicationForm__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.setTo('Administration_form_and_student_record_data_sc_af_tradru');
25    }
}
Org has been successfully connected. No LWC component found.
```

College Management Application

Day 11:

Topic : **College Management Application.**

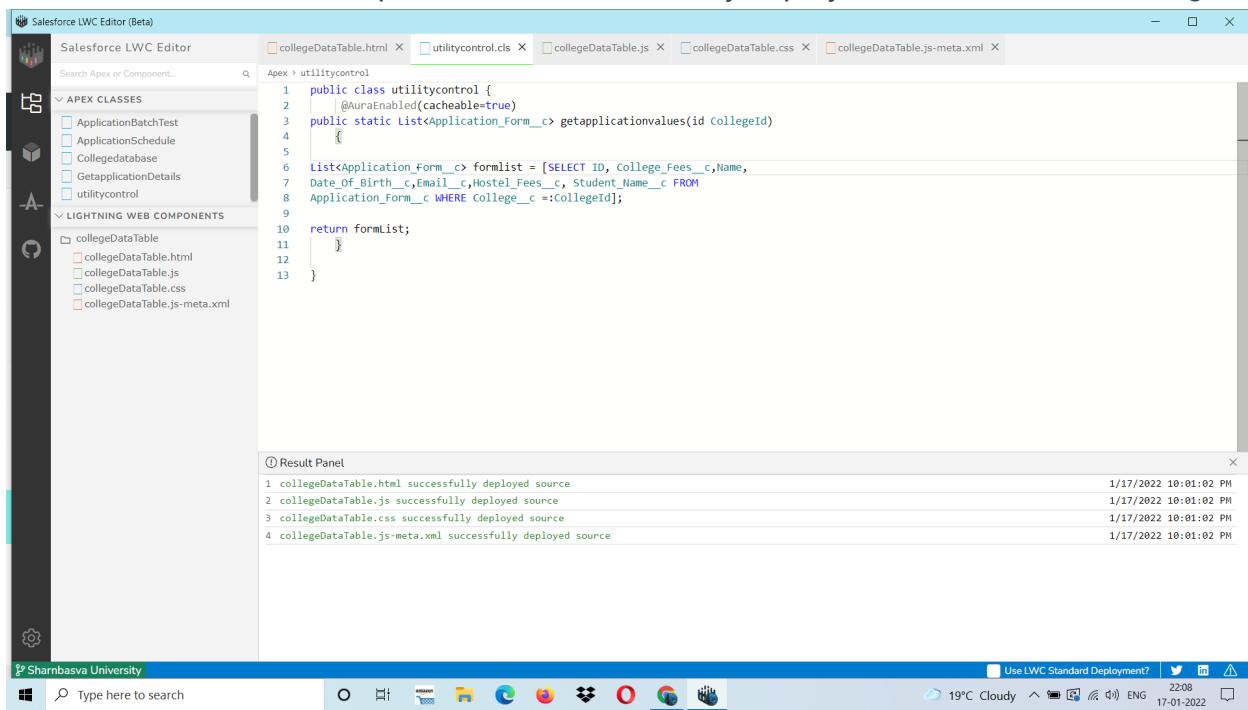
Milestone/Activities : Create College DataTable Component(APEX CLASS)

Detailed Description : In Salesforce LWC Editor created the APEX class named as Utilitycontrol , this apex class is helpful in creating the college datatable where we can call the college Datatable, college datatable consist of one html file and one js file and one css file and also one meta file. this apex class is helpful in calling the college datatable lightning web component.

```
public class GetapplicationDetails
{
    @AuraEnabled(cacheable=true)
    public static
    List<ApplicationForm__c> getapplicationvalues(id CollegeId)
    {
        List<ApplicationForm__c> formlist = [SELECT ID, College_Fees__c, Name,
DateOfBirth__c, Email__c, Hostel_Fees__c, Student_Name__c FROM
ApplicationForm__c
WHERE College__c =:CollegeId];
        return formList;
    }
}
```

College Management Application

ScreenShot : created the apex class and successfully deployed it to the salesforce org.



The screenshot shows the Salesforce LWC Editor (Beta) interface. On the left, the sidebar displays the project structure under "APEX CLASSES" and "LIGHTNING WEB COMPONENTS". The main editor area contains the code for the "utilitycontrol.cls" Apex class:

```
1 public class utilitycontrol {
2     @AuraEnabled(cacheable=true)
3     public static List<Application_Form__c> getapplicationvalues(id CollegeId)
4     {
5
6         List<Application_Form__c> formList = [SELECT ID, College_Fees__c,Name,
7 Date_of_Birth__c,Email__c,Hostel_Fees__c, Student_Name__c FROM
8 Application_Form__c WHERE College__c =:CollegeId];
9
10    return formList;
11 }
12
13 }
```

Below the editor is the "Result Panel" which lists the deployment history:

File	Action	Date
collegeDataTable.html	successfully deployed source	1/17/2022 10:01:02 PM
collegeDataTable.js	successfully deployed source	1/17/2022 10:01:02 PM
collegeDataTable.css	successfully deployed source	1/17/2022 10:01:02 PM
collegeDataTable.js-meta.xml	successfully deployed source	1/17/2022 10:01:02 PM

The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray.

College Management Application

Day 12:

Topic : **College Management Application.**

Milestone/Activities : Created the LWC Component collegeDataTable

- 1)Create College DataTable Component (HTML FILE)
- 2)Create ColleDataTable Component(JAVA SCRIPT FILE)
- 3)Create College DataTable Component(META FILE)

In Salesforce LWC Editor created the LWC Component named as CollegeDatatable where it consist of four files they are one is html file and one is css file and another one is javascript file and the last one is meta file and this files are included in the lightning page , app page and also in the home page. this all pages are called by the apex class .

we successfully saved each file and deployed it in the salesforce org .

College Management Application

ScreenShot : created the collegeDataTable and deployed to the org(HTML FILE)

The screenshot shows the Salesforce LWC Editor interface. On the left, there's a sidebar with icons for Apex Classes, Lightning Web Components, and Utilities. Under 'LIGHTNING WEB COMPONENTS', a component named 'collegeDataTable' is selected, showing its files: collegeDataTable.html, collegeDataTable.js, collegeDataTable.css, and collegeDataTable.js-meta.xml. The main editor area displays the code for collegeDataTable.html, which includes an

tag and a lightning-datatable component. Below the editor is a 'Result Panel' showing deployment logs:

Log Message	Date
1 collegeDataTable.html successfully deployed source	1/17/2022 10:01:02 PM
2 collegeDataTable.js successfully deployed source	1/17/2022 10:01:02 PM
3 collegeDataTable.css successfully deployed source	1/17/2022 10:01:02 PM
4 collegeDataTable.js-meta.xml successfully deployed source	1/17/2022 10:01:02 PM

The bottom of the screen shows a Windows taskbar with various pinned icons and the date/time: 17-01-2022.

ScreenShot : Create ColleDataTable Component(JAVA SCRIPT FILE)

College Management Application

The screenshot shows the Salesforce LWC Editor interface. On the left, the sidebar lists components under 'APEX CLASSES' and 'LIGHTNING WEB COMPONENTS'. In the center, the code editor displays the 'collegeDataTable.js' file:

```
1 import { LightningElement, api, wire } from 'lwc';
2 import getapplicationvalues;
3 import {@salesforce/apex/utilitycontrol.getapplicationvalues};
4 export default class CollegeDataTable extends LightningElement {
5     columnList = [
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: 'Student Name', 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        }
26    }
}
```

The 'Result Panel' at the bottom shows deployment logs:

File	Deployment Time
collegeDataTable.html	1/17/2022 10:01:02 PM
collegeDataTable.js	1/17/2022 10:01:02 PM
collegeDataTable.css	1/17/2022 10:01:02 PM
collegeDataTable.js-meta.xml	1/17/2022 10:01:02 PM

The browser taskbar at the bottom indicates the application is running on Shambava University.

ScreenShot : Create College DataTable Component(META FILE)

The screenshot shows the Salesforce LWC Editor interface. On the left, the sidebar lists components under 'APEX CLASSES' and 'LIGHTNING WEB COMPONENTS'. In the center, the code editor displays the 'collegeDataTable.js-meta.xml' file:

```
<?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 'Result Panel' at the bottom shows deployment logs:

File	Deployment Time
collegeDataTable.html	1/17/2022 10:01:02 PM
collegeDataTable.js	1/17/2022 10:01:02 PM
collegeDataTable.css	1/17/2022 10:01:02 PM
collegeDataTable.js-meta.xml	1/17/2022 10:01:02 PM

The browser taskbar at the bottom indicates the application is running on Shambava University.

College Management Application