

# PROJECT REPORT



SALESFORCE  
PROJECTREADY



**SALESFORCE PROJECT READY**  
**Entry Level Training & Professional Development**  
**Program**

**PROJECT TITLE - COLLEGE MANAGEMENT  
APPLICATION**

# **PROJECT REPORT**

Day 1 :

Topic: Introduction to the course, Cloud Computing Basics, Service Models (IaaS, PaaS, SaaS), Salesforce basics and opportunities, Introduction to objects, fields, relationships. Types of Sandbox (Developer, Developer Pro, Partial, Full), Project specification and allocation of work.

Milestone / Activities: Create Your Salesforce Developer Org To Get Started.Creating Developer Account, Account Activation, Login To Your Salesforce Account.

Detailed Description:

Creating a developer org in salesforce

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

1. Click on sign up.
2. On the sign up form, enter the following details :
  1. First name & Last name
  2. Email
  3. Role : Developer
  4. Company : College Name
  5. County : India
  6. Postal Code : pin code
  7. Username : should be a combination of your name and company

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

username@organization.com

Click on sign up after filling these.

Account Activation

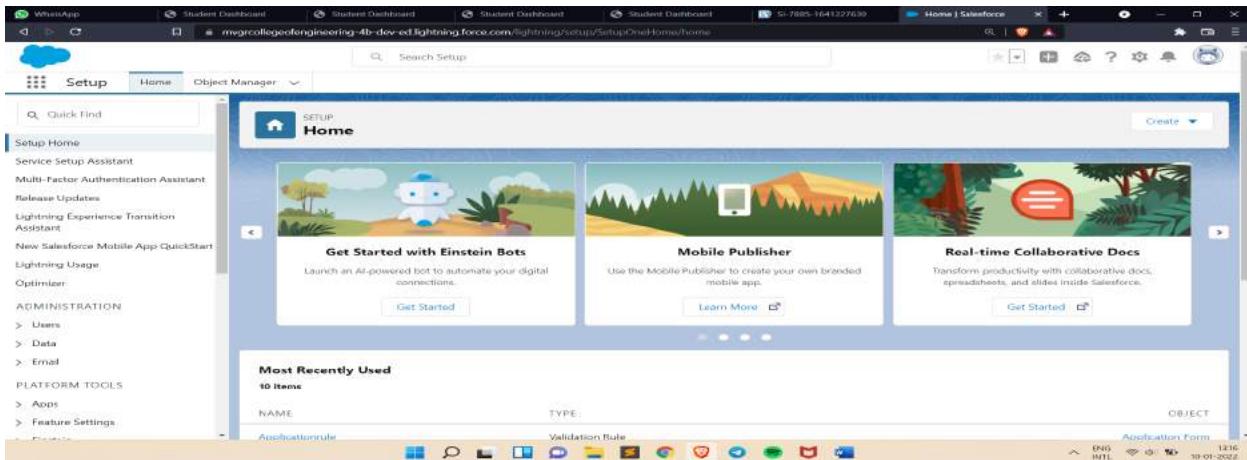
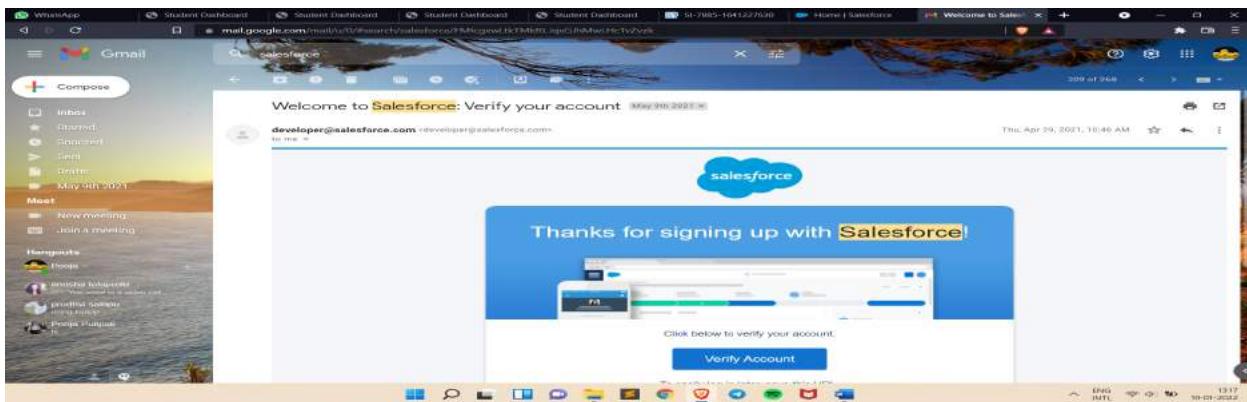
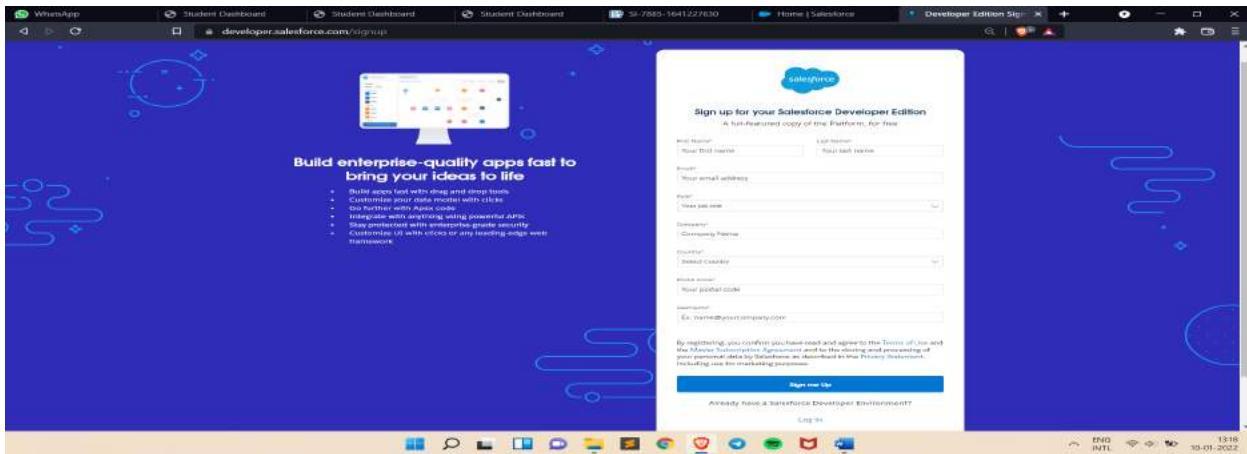
Go to the inbox of the email that you used while signing up. Click on the verify account to activate your account. The email may take 5-10mins, as allocating a developer org will take time.

Login To Your Salesforce Account

- 1.Go to [salesforce.com](https://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.

# PROJECT REPORT

Upload the Screenshot the Milestone / Activities :



# PROJECT REPORT

Day 2 :

Topic: Custom Objects and their creation, Adding business logic to the application.

Milestone / Activities: Create The Custom Objects For The Application, Create Fields On College Object, Create Fields On Application Form Object, Create Fields On Student Object, Create Fields On Subject Object. Creating Global Picklist Value Sets, Creating Field Dependencies.

Detailed Description:

## Creating custom objects

Go to the object manager tab and click it. There we can create custom objects by clicking new custom object. For all the four objects ie. College, Application Form, Student and Subject, create them as objects with given labels. The Next step is to create respective fields for all the four custom objects with some data. Creating fields involve data types to be chosen and can also choose whether the field is meant to be required.

## Setting global picklist values

Global picklist values can be added to any objects. Edit button is used and picklist type data is chosen and values can be added.

## Creating field dependencies

In object manager, click on objects and go to field dependencies for creating field dependencies. Create field dependency between the college Name and Email, where the controlling field is the college Name and dependent field is Email. Select the email ids according to the college names. Create field dependency between college Name and capacity of students, where the controlling field is college Name and dependant field is Capacity of Students. Select the values according to your wish.

# PROJECT REPORT

Upload the Screenshot the Milestone / Activities :

This screenshot shows the Salesforce Object Manager interface for the 'College' object. The page title is 'SETUP > OBJECT MANAGER College'. The left sidebar lists various configuration categories such as Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Triggers, and Validation Rules. The main content area is titled 'Fields & Relationships' and displays 10 items sorted by Field Label. The table includes columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. Key fields shown include Capacity Of Students, College Fees, College ID, College Name, Created By, Email, Hostel Fees, Last Modified By, and Owner.

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 ID	College_ID_c	Auto Number		
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(9, 2)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User/Group)		

This screenshot shows the Salesforce Object Manager interface for the 'Application Form' object. The page title is 'SETUP > OBJECT MANAGER Application Form'. The left sidebar lists various configuration categories. The main content area is titled 'Fields & Relationships' and displays 13 items sorted by Field Label. The table includes columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. Key fields shown include Address, Application Form ID, Application Form Name, College, College Fees, Created By, Date of Birth, Email, Guardian Name, Hostel Fee, Last Modified By, Looking For Hostel Stay, Phone, Ready To Join, and Student Name.

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address_c	Text Area(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)		
Created By	CreatedById	Lookup(User)		
Date of Birth	Date_of_Birth_c	Date		
Email	Email_c	Email (Unique)		
Guardian Name	Guardian_Name_c	Text(20)		
Hostel Fee	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)		

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface for the 'Student' object. The top navigation bar includes tabs for WhatsApp, Student Dashboard, Object Manager, and a specific record view. The main header displays 'mvrgcollegeofengineering-4b-dev-ed.lightning.force.com/lightning/setup/ObjectManager/015g00000210WE/FieldsAndRelationships/view'. A search bar labeled 'Search Setup' is at the top right. Below the header, a sidebar on the left lists various setup categories like Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Triggers, and Validation Rules. The main content area is titled 'Fields & Relationships' and shows 11 items sorted by Field Label. It includes columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The fields listed are:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Address	Address_c	Text Area(255)		
Application Form	Application_Form_c	Lookup(Application Form)		✓
College Name	College_Name_c	Formula (Text)		
Created By	CreatedById	Lookup(User)		
Date Of Birth	DateOfBirth_c	Date		
Guardian Name	Guardian_Name_c	Text(30)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
Phone	Phone_c	Phone		
Student Name	Student_Name_c	Text(254)		
Students Name	Name	Text(80)		✓

The bottom status bar shows system icons and the date/time: 13:57 10-01-2022.

The screenshot shows the Salesforce Object Manager interface for the 'Subject' object. The top navigation bar and header are identical to the previous screenshot. The main content area is titled 'Fields & Relationships' and shows 8 items sorted by Field Label. It includes columns for FIELD LABEL, FIELD NAME, DATA TYPE, CONTROLLING FIELD, and INDEXED. The fields listed are:

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		
Subjects Name	Name	Text(80)		✓

The bottom status bar shows system icons and the date/time: 13:57 10-01-2022.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface. The left sidebar lists various setup categories like Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, etc. The main content area displays the 'Custom Field Definition Detail' for a field named 'Page1'. The 'Field Information' section shows the Field Label as 'Page1', Field Name as 'Page1', and API Name as 'Page1\_L1'. It also includes fields for Description, Help Text, Data Owner, and Field Usage. The 'Data Sensitivity Level' is set to 'Compliance Configuration'. The 'Created By' field shows 'Esha.Mathuraa.Sinha' and the 'Modified By' field shows 'Esha.Mathuraa.Sinha'. The 'General Options' section includes 'Required' (unchecked) and 'Default Value' (empty). The 'Picklist Options' section shows a single option 'Restrict picklist to the values defined in the value set' with 'Controlling Field' set to 'Page1'. The 'Picklist Values Used' section shows one value 'Active and inactive picklist values: 4 (1,30 mri)'. The 'Field Dependencies' section is empty. The 'Validation Rules' section is empty. The 'Values' section contains four entries:

Action	Value	API Name	Default	Chart Colors	Modified By
Edit	Java	Java	Assigned dynamically		Esha.Mathuraa.Sinha, 14/02/22, 4:39 AM
Edit	JavaScript	JS	Assigned dynamically		Esha.Mathuraa.Sinha, 14/02/22, 4:39 AM
Edit	C	C	Assigned dynamically		Esha.Mathuraa.Sinha, 14/02/22, 4:39 AM
Edit	C++	Cpp	Assigned dynamically		Esha.Mathuraa.Sinha, 14/02/22, 4:39 AM

The 'Inactive Values' section is empty.

This screenshot shows the same Salesforce Object Manager interface, but the custom field 'Page1' has been renamed to 'Page2'. The 'Field Information' section now shows the Field Label as 'Page2', Field Name as 'Page2', and API Name as 'Page2\_L1'. The rest of the field definition remains identical to the first screenshot, including the picklist options, validation rules, and value list.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface for the 'Subject' object. The 'Fields & Relationships' tab is selected. A custom field named 'Student' is being edited. The 'Field Information' section shows the field label 'Student', field name 'Student', API name 'Student\_\_c', and data type 'Lookup'. The 'Object Name' is 'Subject'. The 'Related To' field is set to 'Student'. The 'Child Relationship Name' is 'Subjects'. The 'Created By' and 'Modified By' fields show 'Popja Madhumita Saha'. The 'Validation Rules' section indicates 'No validation rules defined'. The bottom status bar shows the date as 10-01-2022.

The screenshot shows the Salesforce Object Manager interface for the 'Subject' object. The 'Fields & Relationships' tab is selected. A custom field named 'Subject ID' is being edited. The 'Field Information' section shows the field label 'Subject ID', field name 'Subject\_ID', API name 'Subject\_ID\_c', and data type 'Auto Number'. The 'Object Name' is 'Subject'. The 'General Options' section shows 'External ID' is disabled. The 'Auto Number Options' section shows the 'Display Format' as 'S-{000000}'. The bottom status bar shows the date as 10-01-2022.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface. The left sidebar is titled "Fields & Relationships" and includes links for Details, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Triggers, and Validation Rules. The main content area is titled "Custom Field Definition Detail" for the "Address" field of the "Student" object. The "Field Information" section shows the field label as "Address", field name as "Address", API name as "Address\_\_c", and data type as "Text Area". The "General Options" section indicates that the field is required. The "Validation Rules" section shows a note: "No validation rules defined". The bottom right corner of the window shows the date and time as 10-01-2022, 10:22 AM.

The screenshot shows the Salesforce Object Manager interface. The left sidebar is identical to the previous screenshot. The main content area is titled "Custom Field Definition Detail" for the "Application\_Form" field of the "Student" object. The "Field Information" section shows the field label as "Application Form", field name as "Application\_Form", API name as "Application\_Form\_\_c", and data type as "Lookup". The "Lookup Options" section shows the related to object as "Application Form" and the child relationship name as "Students". The "Validation Rules" section shows a note: "No validation rules defined". The bottom right corner of the window shows the date and time as 10-01-2022, 10:22 AM.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface. A custom field named 'College Name' is being created for the 'Student' object. The field has a label 'College Name', a name 'College\_Name', and an API name 'College\_Name\_c'. It is a formula type field with the formula 'Student\_\_Name\_\_c'. The field is owned by 'Esha Mardhuria Saloni' and was created on 14/02/2022 at 4:20 AM.

**Custom Field Definition Detail**

Field Information	Object Name
Field Label: College Name	Object Name: Student
Field Name: College_Name	data type: Text
API Name: College_Name_c	
Description:	
Help Text:	
Data Owner:	Esha Mardhuria Saloni
Field Usage:	
Data Sensitivity Level:	
Compliance Categorization:	
Created By: Esha Mardhuria Saloni	Created Date: 14/02/2022, 4:20 AM
Modified By: Esha Mardhuria Saloni	Modified Date: 14/02/2022, 4:20 AM

The screenshot shows the Salesforce Object Manager interface. A custom field named 'Date Of Birth' is being created for the 'Student' object. The field has a label 'Date Of Birth', a name 'Date\_of\_Birth', and an API name 'Date\_of\_Birth\_c'. It is a date type field with a required status. The field is owned by 'Esha Mardhuria Saloni' and was created on 14/02/2022 at 4:20 AM.

**Custom Field Definition Detail**

Field Information	Object Name
Field Label: Date Of Birth	Object Name: Student
Field Name: Date_of_Birth	data type: Date
API Name: Date_of_Birth_c	
Description:	
Help Text:	
Data Owner:	Esha Mardhuria Saloni
Field Usage:	
Data Sensitivity Level:	
Compliance Categorization:	
Created By: Esha Mardhuria Saloni	Created Date: 14/02/2022, 4:20 AM
Modified By: Esha Mardhuria Saloni	Modified Date: 14/02/2022, 4:22 AM

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface for the 'Student' object. The left sidebar contains navigation links such as Setup, Home, Object Manager, Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Triggers, and Validation Rules. The main content area displays the 'Custom Field Definition Detail' for the 'Guardian Name' field. The field information includes:

Field Label	Guardian Name	Object Name	Student
Field Name	Guardian_Name	Data Type	Text
API Name	Guardian_Name__c		
Description			
Help Text			
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			

General Options include Required (checked), Unique, Case Sensitive, External ID, and Default Value. Text Options show a Length of 30. Validation Rules state 'No validation rules defined'. The page also includes a 'Validation Rules Help' link and a note to 'Always show me ▾ more records per related list'. The bottom of the screen shows the Windows taskbar with various pinned icons and system status.

The screenshot shows the Salesforce Object Manager interface for the 'Student' object. The left sidebar contains the same navigation links as the previous screenshot. The main content area displays the 'Custom Field Definition Detail' for the 'Phone' field. The field information includes:

Field Label	Phone	Object Name	Student
Field Name	Phone	Data Type	Phone
API Name	Phone__c		
Description			
Help Text			
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			

General Options include Required (checked), Unique, Case Sensitive, External ID, and Default Value. Validation Rules state 'No validation rules defined'. The page also includes a 'Validation Rules Help' link and a note to 'Always show me ▾ more records per related list'. The bottom of the screen shows the Windows taskbar with various pinned icons and system status.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface for the 'Student' object. The 'Fields & Relationships' tab is selected, displaying the 'Student Name' field. The field details include:

- Field Label:** Student Name
- Field Name:** Student\_Name
- API Name:** Student\_\_Name\_c
- Description:** Help Text
- Data Owner:** Field Usage
- Data Sensitivity Level:** Compliance Categorization
- Object Name:** Student
- Data Type:** Text

The 'General Options' section shows 'Required' and 'Unique' checkboxes checked. The 'Text Options' section shows a 'Length' of 254. The 'Validation Rules' section is currently empty.

The screenshot shows the Salesforce Object Manager interface for the 'College' object. The 'Fields & Relationships' tab is selected, displaying the 'Capacity Of Students' field. The field details include:

- Field Label:** Capacity Of Students
- Field Name:** Capacity\_\_c
- API Name:** Capacity\_\_c
- Description:** Help Text
- Data Owner:** Field Usage
- Data Sensitivity Level:** Compliance Categorization
- Object Name:** College
- Data Type:** Number

The 'General Options' section shows 'Required' and 'Default Value' checkboxes checked. The 'Picklist Options' section shows a picklist value 'Capacity' assigned to the value set 'Capacity'. The 'Picklist Values Used' section shows a single value 'Auto and inactive picklist values (1,122 total)'. The 'Field Dependencies' section shows no dependencies defined. The 'Validation Rules' section is currently empty.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface. The left sidebar lists various setup categories like Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Triggers, and Validation Rules. The main content area displays the 'Custom Field Definition Detail' for the 'College' object. The 'Field Information' section includes fields such as Field Label (College Name), Field Name (College\_Name), API Name (College\_Name\_\_c), Description, Help Text, Data Type (Text), Data Length (255), and Data Precision (0). The 'General Options' section shows 'Required' checked and 'Default Value' set to null. The 'Picklist Options' section indicates that restrict picklist values is checked, with 'Controlling Field' set to 'Name'. The 'Picklist Values Used' section shows a limit of 2 (100 max). The 'Field Dependencies' section lists dependencies for the 'College' object's 'Name' field, with actions like 'Edit' and 'Delete' mapped to 'MTC01' and 'MTC02'. The 'Validation Rules' section is currently empty. The 'Values' section shows a table of values for the 'Hostel\_Fees' field, with rows for 'MTC01' through 'MTC05'. The top right of the page shows the URL <https://mvgrcollegeofengineering-4b-dev-ed.lightning.force.com/lightning/setup/ObjectManager/0115g00000210W4/FieldsAndRelationships/00N5g00000...>, and the bottom right shows system status icons.

The screenshot shows the Salesforce Object Manager interface, similar to the previous one but for the 'Hostel Fees' field. The left sidebar is identical. The main content area displays the 'Custom Field Definition Detail' for the 'Hostel Fees' field. The 'Field Information' section includes fields such as Field Label (Hostel Fees), Field Name (Hostel\_Fees), API Name (Hostel\_Fees\_\_c), Description, Help Text, Data Owner, and Data Usage. The 'General Options' section shows 'Required' checked and 'Default Value' set to null. The 'Currency Options' section shows a length of 6 and decimal places of 2. The 'Validation Rules' section is currently empty. The top right of the page shows the URL <https://mvgrcollegeofengineering-4b-dev-ed.lightning.force.com/lightning/setup/ObjectManager/0115g00000210W4/FieldsAndRelationships/00N5g00000...>, and the bottom right shows system status icons.

# PROJECT REPORT

The screenshot shows the 'Application Form' custom field configuration in the Salesforce Setup. The 'Address' field is selected. The 'Field Information' section shows the field label 'Address', field name 'Address', and API name 'Address\_\_s'. The object name is 'Application Form' and the data type is 'Text Area'. The 'General Options' section has 'Required' checked. There are no validation rules defined.

The screenshot shows the 'Application Form ID' custom field configuration in the Salesforce Setup. The field label is 'Application Form ID', field name is 'Application\_Form\_ID', and API name is 'Application\_Form\_ID\_\_s'. The object name is 'Application Form' and the data type is 'Auto Number'. The 'General Options' section shows 'External ID' selected. The 'Auto Number Options' section shows a display format of 'F-00000'.

# PROJECT REPORT

This screenshot shows the Salesforce Object Manager interface for the 'Application Form' object. The left sidebar contains navigation links such as Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Triggers, and Validation Rules. The main content area displays the 'Application Form Field' details for the 'Application Form Name' field. The 'Field Information' section includes the field label ('Application Form Name'), data type ('Text(400)'), description (''), data owner (''), field usage (''), and data sensitivity level ('Compliance Categorization'). The 'Validation Rules' section indicates 'No validation rule defined.' A status bar at the bottom right shows 'Battery status: 73% available (plugged in)' and the system date/time '10-01-2022 10:53'.

This screenshot shows the Salesforce Object Manager interface for a custom field named 'College' on the 'Application Form' object. The left sidebar is identical to the previous screenshot. The main content area displays the 'Custom Field Definition Detail' for the 'College' field. The 'Field Information' section includes the field label ('College'), field name ('College'), API name ('College\_\_c'), description (''), help text (''), data owner (''), field usage (''), and data sensitivity level ('Compliance Categorization'). The 'Object Name' is 'Application Form' and the 'Data Type' is 'Master-Detail'. The 'Master-Detail Options' section shows 'Related To' as 'College', 'Related List Label' as 'Applicant Forms', and 'Sharing Setting' as 'Read/Write. Allows users with at least Read/Write access to the Master record to create, edit, or delete related Detail records.' The 'Master Relationship Name' is 'Application\_Forms'. The 'Validation Rules' section indicates 'No validation rule defined.' A status bar at the bottom right shows 'Show desktop', 'Battery status: 73% available (plugged in)', and the system date/time '10-01-2022 10:53'.

# PROJECT REPORT

The screenshot shows the Salesforce Setup interface for the 'Application Form' object. A specific custom field, 'College Fees', is being edited. The 'Fields & Relationships' tab is selected. The 'Field Information' section displays details such as Field Label ('College Fees'), Field Name ('College\_Fees'), and API Name ('College\_Fees\_\_c'). The 'Data Type' is set to 'Formula'. The formula definition is: `1+1`. Other tabs visible include 'Page Layouts', 'Lightning Record Pages', and 'Validation Rules'.

The screenshot shows the Salesforce Setup interface for the 'Application Form' object. A specific custom field, 'date of Birth', is being edited. The 'Fields & Relationships' tab is selected. The 'Field Information' section displays details such as Field Label ('date of Birth'), Field Name ('date\_of\_Birth'), and API Name ('date\_of\_Birth\_\_c'). The 'Data Type' is set to 'Date'. The 'General Options' section shows 'Required' checked. The 'Validation Rules' section indicates 'No validation rules defined'. Other tabs visible include 'Page Layouts', 'Lightning Record Pages', and 'Validation Rules'.

# PROJECT REPORT

The screenshot shows the Salesforce Setup interface under the Object Manager. A custom field named 'Email' is being edited for the 'Application Form' object. The 'Fields & Relationships' tab is selected. The 'Field Information' section shows the field label 'Email', API name 'Email\_\_c', and data type 'Email'. The 'General Options' section has 'Unique' checked. The 'Validation Rules' section indicates 'No validation rules defined'. The top right shows the object name 'Application Form' and data type 'Email'. The bottom right shows the date '10-01-2022' and time '10:53'.

The screenshot shows the Salesforce Setup interface under the Object Manager. A custom field named 'Guardian Name' is being edited for the 'Application Form' object. The 'Fields & Relationships' tab is selected. The 'Field Information' section shows the field label 'Guardian Name', API name 'Guardian\_Name\_\_c', and data type 'Text'. The 'General Options' section has 'Unique' checked. The 'Text Options' section shows a length of '10'. The 'Validation Rules' section indicates 'No validation rules defined'. The top right shows the object name 'Application Form' and data type 'Text'. The bottom right shows the date '10-01-2022' and time '10:53'.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface. The left sidebar lists various setup options like Fields & Relationships, Page Layouts, Lightning Record Pages, etc. The main content area displays the 'Hostel Fees' custom field definition for the 'Application Form' object. The field information includes:

- Field Label:** Hostel Fees
- Field Name:** Hostel\_Fee
- API Name:** Hostel\_Fee\_\_c
- Description:** Hostel Fee
- Data Type:** Number
- Field Usage:** Standard
- Data Sensitivity Level:** Standard
- Compliance Category:** None
- Created By:** Data Mathematics Setup (14/02/2022 4:07 AM)
- Modified By:** Data Mathematics Setup (14/02/2022 4:07 AM)

Below this, there are sections for Formula Options (Data Type: Formula, Decimal Places: 2) and Relationship Options (1:1).

The screenshot shows the Salesforce Object Manager interface. The left sidebar lists various setup options like Fields & Relationships, Page Layouts, Lightning Record Pages, etc. The main content area displays the 'Looking For Hostel Stay' custom field definition for the 'Application Form' object. The field information includes:

- Field Label:** Looking For Hostel Stay
- Field Name:** Looking\_For\_Hostel\_Stay
- API Name:** Looking\_For\_Hostel\_Stay\_\_c
- Description:** Looking For Hostel Stay
- Data Type:** Checkbox
- Field Usage:** Standard
- Data Sensitivity Level:** Standard
- Compliance Category:** None
- Created By:** Data Mathematics Setup (14/02/2022 4:09 AM)
- Modified By:** Data Mathematics Setup (14/02/2022 4:09 AM)

Below this, there are sections for General Options (Default Value: Unchecked), Field Dependencies (No dependencies defined), and Validation Rules (No validation rules defined). A note at the bottom says "Assign this record to one or more records per related list".

# PROJECT REPORT

The screenshot shows the Salesforce Setup - Object Manager interface. A new custom field is being created for the 'Application Form' object. The field is named 'Phone' and has the API name 'Phone\_\_c'. It is a Phone type field with a length of 15. The field is marked as required. There are no validation rules or field dependencies defined.

**Custom Field Definition Detail**

**Field Information**

- Field Label: Phone
- Field Name: Phone
- API Name: Phone\_\_c
- Description:
- Help Text:
- Data Type: Phone
- Field Usage:
- Date Sensitivity Level:
- Compliance Categorization:
- Created By: Poornima Suleja (14/02/2022, 4:19 AM)
- Modified By: Poornima Suleja (14/02/2022, 4:19 AM)

**General Options**

- Required:
- Default Value:

**Validation Rules**

No validation rules defined.

Always show one ▾ more records per related list

The screenshot shows the Salesforce Setup - Object Manager interface. A new custom field is being created for the 'Application Form' object. The field is named 'Ready To Join' and has the API name 'Ready\_To\_Join'. It is a Checkbox type field. The field is not marked as required. There are no validation rules or field dependencies defined.

**Custom Field Definition Detail**

**Field Information**

- Field Label: Ready To Join
- Field Name: Ready\_To\_Join
- API Name: Ready\_To\_Join\_\_c
- Description:
- Help Text:
- Data Type: Checkbox
- Field Usage:
- Date Sensitivity Level:
- Compliance Categorization:
- Created By: Poornima Suleja (14/02/2022, 4:09 AM)
- Modified By: Poornima Suleja (14/02/2022, 4:09 AM)

**General Options**

- Default Value: Unchecked

**Field Dependencies**

No dependencies defined.

**Validation Rules**

No validation rules defined.

Always show one ▾ more records per related list

# PROJECT REPORT

The screenshot shows the Salesforce Setup interface for the Application Form object. The left sidebar lists various setup categories like Details, Fields & Relationships, Page Layouts, etc. The main content area is titled 'Custom Field Definition Detail' for the 'Student Name' field. It shows the field label 'Student Name', field name 'Student\_Name', and data type 'Text'. Other details include 'Required' checked, 'Unique' unchecked, and 'Case Sensitive' checked. Validation rules and field dependencies are also listed.

The screenshot shows the Salesforce Setup interface for the College object. The left sidebar lists various setup categories. The main content area is titled 'General Options' for the 'Capacity of Students' field. It shows 'Required' checked and 'Default Value' as '1'. Under 'Field Dependencies', it lists 'Dependent Field' as 'Email' and 'Moored By' as 'Pooya.Madhurima.Salapu'. Validation rules are listed as 'No validation rules defined'.

# PROJECT REPORT

Day 3 :

Topic: Adding Business Logic To Application, Automations.

Milestone / Activities: Creating Validation Rules, Process Automation, Create The Student Record Using Flow.

Detailed Description:

## Create validation rule for objects

Validation rule on the college object such that the college name and record info should have the same name. Click on an object and select the validation rule option. Add rule without syntax and save it.

## Create process

For the process automation, Go to Setup -> select "Process Builder" from quick find. Create a Process Builder on the "Application Form" object with a condition as "When a record change". And select "When a record is created or edited". In the diamond shape box(called nodes), select the criteria which trigger the Process builder to fire. In our example, it is "When Ready to Join field is checked. Once the node is set up, click on the adjacent box called "Immediate action". And select create a record on the student object.

## Create flows

First deactivate the process builder which we created earlier. Now search for flows and select new flow ->record triggered flow. 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. Now create a variable named student in the resource section. Now add assignments which is mapping from student fields to application fields. Now add create records. Create records is dragged to flow area and create a student record is added. Fields are assigned as mapping of student fields to application form fields.

# PROJECT REPORT

Upload the Screenshot the Milestone / Activities :

The screenshot displays two screenshots of the Salesforce Lightning interface.

The top screenshot shows the "Recently Viewed" section for Accounts. The URL is <https://mvgrcollegeofengineering-4b-dev-ed.lightning.force.com/lightning/c/Account/list?filterName=Recent>. The page header includes "College Management" and navigation links for Accounts, Contacts, Colleges, and Application Forms. The main content area displays a message: "You haven't viewed any Accounts recently. Try switching list views." There are search and filter tools at the top.

The bottom screenshot shows the "Validation Rules" section under the "Application Form" object in the Object Manager. The URL is <https://mvgrcollegeofengineering-4b-dev-ed.lightning.force.com/lightning/setup/ObjectManager/0H5g000003210W5/ValidationRules/view>. The page header includes "Setup" and "Object Manager". The main content area lists one validation rule:

RULE NAME	ERROR LOCATION	ERROR MESSAGE	ACTIVE	MODIFIED BY
Applicationrule	Top of Page	Once created, application form can't be modified.	✓	Pooja Madhurima Salapu, 1/6/2022, 5:41 AM

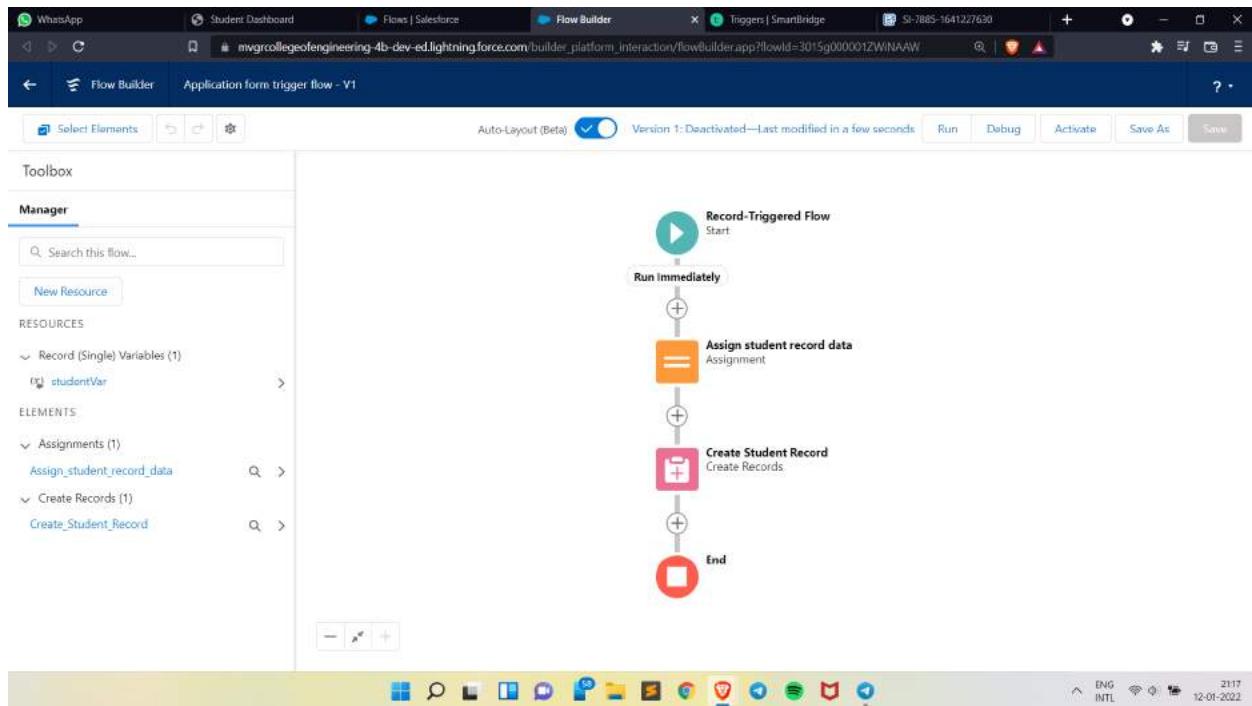
The left sidebar of the bottom screenshot lists various setup categories: Details, Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, and Triggers.

# PROJECT REPORT

The screenshot shows the Salesforce Setup interface for the 'College' object. On the left, a sidebar lists various setup categories like Details, Fields & Relationships, Page Layouts, etc. The main content area is titled 'Validation Rules' and displays one item: 'Collegerule'. The table columns are RULE NAME, ERROR LOCATION, ERROR MESSAGE, ACTIVE, and MODIFIED BY. The rule 'Collegerule' has an error message 'College name and record info is not same' located at 'Top of Page'. It was modified by 'Pooja Madhurima Salapu' on 1/6/2022, 5:39 AM.

The screenshot shows the Salesforce Process Builder interface for the 'Application form process'. The process starts with a 'START' node, followed by an 'Application Form' step. This leads to a decision diamond 'Ready To Join'. If 'TRUE', it branches to an 'IMMEDIATE ACTIONS' section containing 'Create Record' and '+ Add Action', which then leads to a 'STOP' node. If 'FALSE', it branches to another decision diamond '+ Add Criteria', which then leads to another 'IMMEDIATE ACTIONS' section containing '+ Add Action' and a final 'STOP' node. The top navigation bar shows the URL as mvgrcollegeofengineering-4b-dev-ed.lightning.force.com/lightning/setup/ProcessAutomation/home.

# PROJECT REPORT



Day 4 :

Topic: Security and sharing, overview of apex

Milestone / Activities: Security settings for an object, Knowing how to create apex classes.

Detailed Description: For an object, we can set field level and record level security. We can add accessibility depending on groups, public groups or according to roles. Click on home, and open public groups. Create a group where one or more users can be added. The other process is to click on an object and click on the set field accessibility button to change settings. We can assign hierarchical wise security also.

# PROJECT REPORT

Upload the Screenshot the Milestone / Activities:

The screenshot shows the 'Sharing Settings' page in the Salesforce Setup. The 'Sharing Settings' tab is selected under the 'Security' section. The page displays sharing settings for the 'College' object. It includes sections for 'Default Sharing Settings' (Organization-Wide Defaults) and 'Other Settings'. The 'Default Internal Access' for 'College' is set to 'Public Read/Write'. The 'Default External Access' is 'Private'. Under 'Other Settings', 'Secure guest user record access' is checked. The page also includes links for 'Help for this Page' and 'Organization-Wide Defaults Help'.

The screenshot shows the 'Object Manager' page in the Salesforce Setup. The 'College' object is selected. The left sidebar shows navigation options like 'Details', 'Fields & Relationships', and 'Page Layouts'. The main area displays the 'College name' field details, including its field label ('College name') and data type ('Plist'). Below this is a 'Field-Level Security for Profile' table. The table lists various profiles and their visibility and read-only status. Most profiles have 'Visible' checked and 'Read-Only' unchecked. The table includes profiles such as 'Analytics Cloud Integration User', 'Analytics Cloud Security User', 'Contract Manager', 'Cross Org Data Proxy User', 'Custom Marketing Profile', 'Custom Sales Profile', 'Custom Support Profile', 'Force.com - App Subscription User', 'Force.com - Free User', 'Gold Partner User', and 'Identity User'. The page also includes links for 'Help for this Page' and 'Object Manager Help'.

# PROJECT REPORT

The screenshot shows the Salesforce Object Manager interface. The left sidebar lists various setup categories like Details, Fields & Relationships, Page Layouts, etc. The main pane displays a table of user profiles with checkboxes indicating their selection status. The table includes rows for Force.com - App Subscription User, Force.com - Free User, Gold Partner User, Identity User, Marketing User, Minimum Access - Salesforce, Partner App Subscription User, Partner Community Login User, Partner Community User, Read Only, Sales Rep, Silver Partner User, Solution Manager, Standard Platform User, Standard User, System Administrator, Test Profile, and Work.com Only User.

The screenshot shows the Salesforce Public Groups page. The left sidebar has a search bar and navigation links for Users, Public Groups, Feature Settings, and other settings. The main pane shows a table for the 'Student PG' group. The group details are as follows:

Label	Student PG
Group Name	Student_PG
Grant Access Using Hierarchies	<input checked="" type="checkbox"/>
Created By	Pooja.Madhurima.Salapu 1/12/2022 1:43 AM
Modified By	Pooja.Madhurima.Salapu 1/12/2022 1:43 AM

Below the group details, there is a list of users associated with the group:

Name	Type
Security User	User
Pooja.Madhurima.Salapu	User

A tooltip at the bottom right of the screen indicates "MKS417 Salapu Appa Rao Internet access".

# PROJECT REPORT

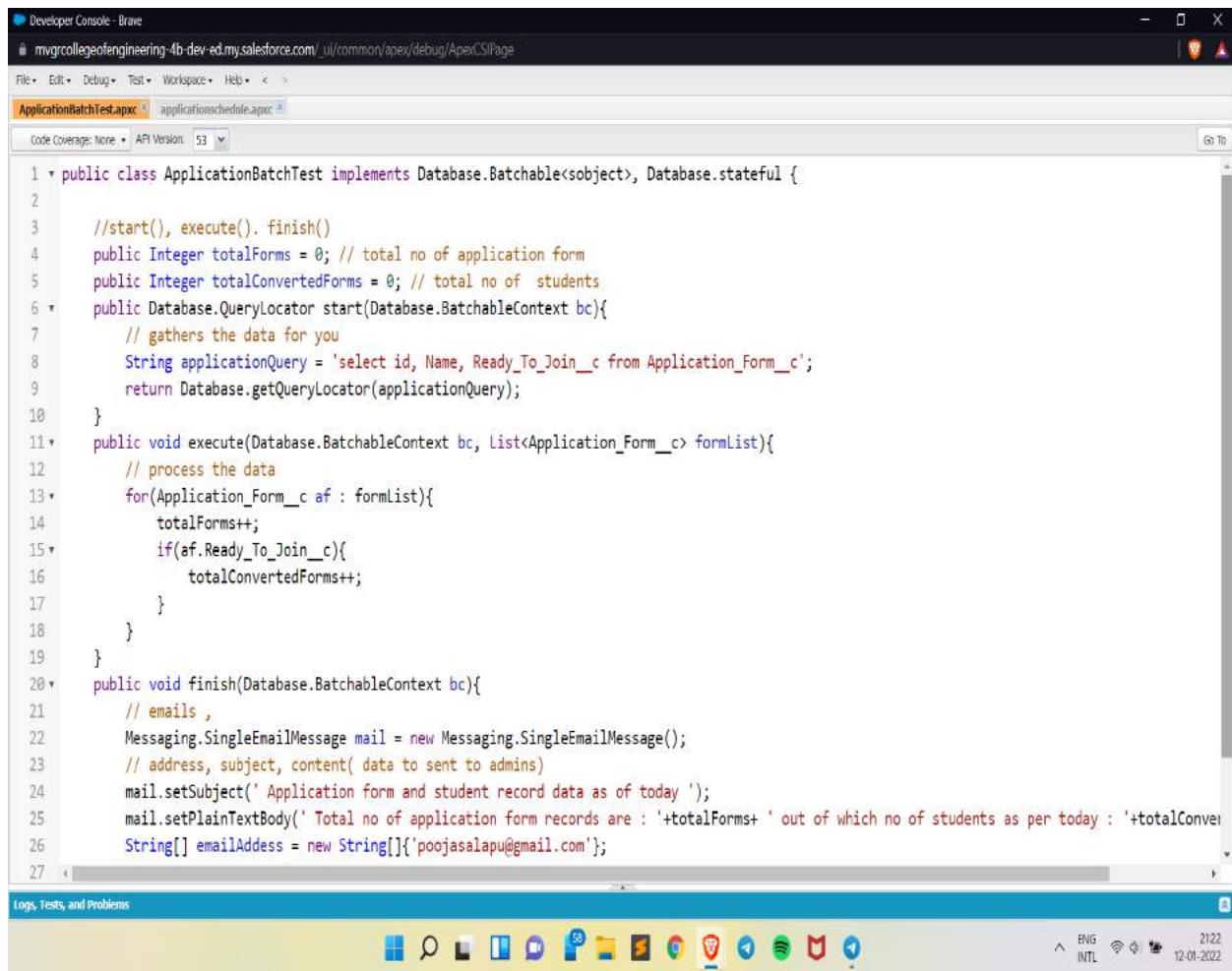
Day 5 :

Topic: Primitive Datatypes, Collections

Milestone / Activities: To know about primitive datatypes, collections

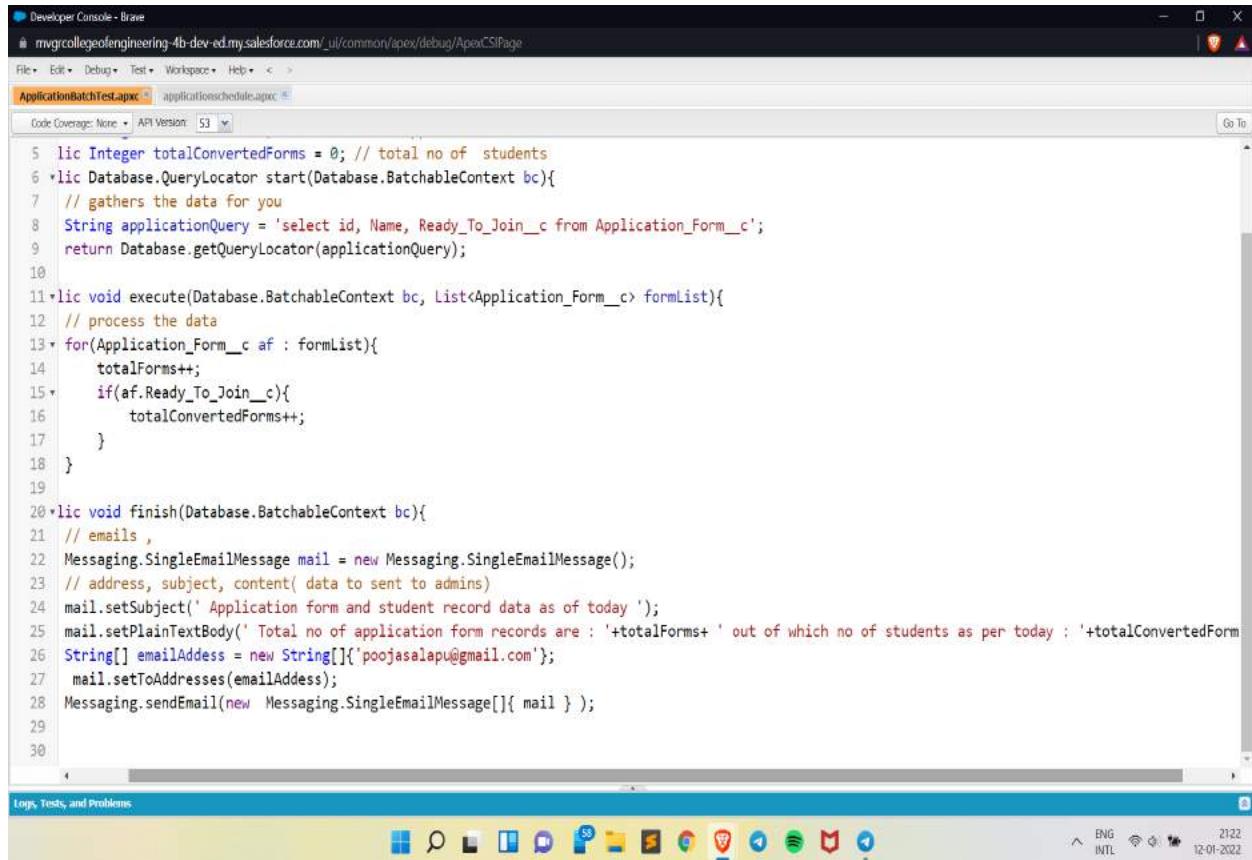
Detailed Description: While using apex, we get to use datatypes like int, boolean, string etc. Collections like List<sobject>, Set<> etc can be used.

Upload the Screenshot the Milestone / Activities :



```
Developer Console - Brave
mvycollegeofengineering-4b-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSPage
File • Edit • Debug • Test • Workspace • Help • < >
ApplicationBatchTest.apxc applicationschedule.apxc
Code Coverage: None • API Version: 53
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[] emailAddress = new String[]{'poojasalapu@gmail.com'};
27    }
}
Logs, Tests, and Problems
Windows Search Taskbar 2122
ENG INTEL 12-01-2022
```

# PROJECT REPORT



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/ApexCSPage". The menu bar includes File, Edit, Debug, Test, Workspace, Help, and a Go To button. The main area displays the following Apex code:

```
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[] emailAddess = new String[]{poojasalapu@gmail.com};
27 mail.setToAddresses(emailAddess);
28 Messaging.sendEmail(new Messaging.SingleEmailMessage[]{ mail } );
29
30
```

The status bar at the bottom shows "Logs, Tests, and Problems" and a toolbar with various icons. The system tray indicates "ENG INTL" and the date "12-01-2022".

Day 6 : Overview of triggers, Batch Apex classes.

Topic:

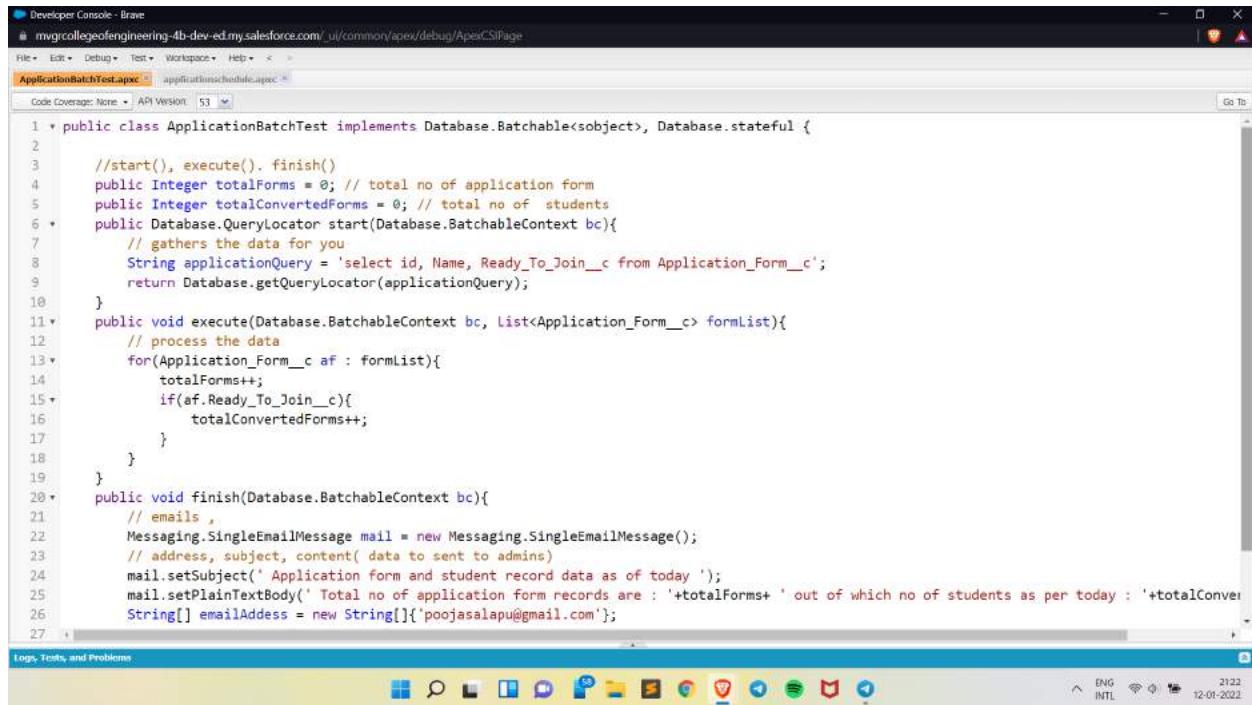
Milestone / Activities: Create A Batch apex For Application Form, Create A Schedular Class

Detailed Description: Open developer console and using new option we can create new apex classes.

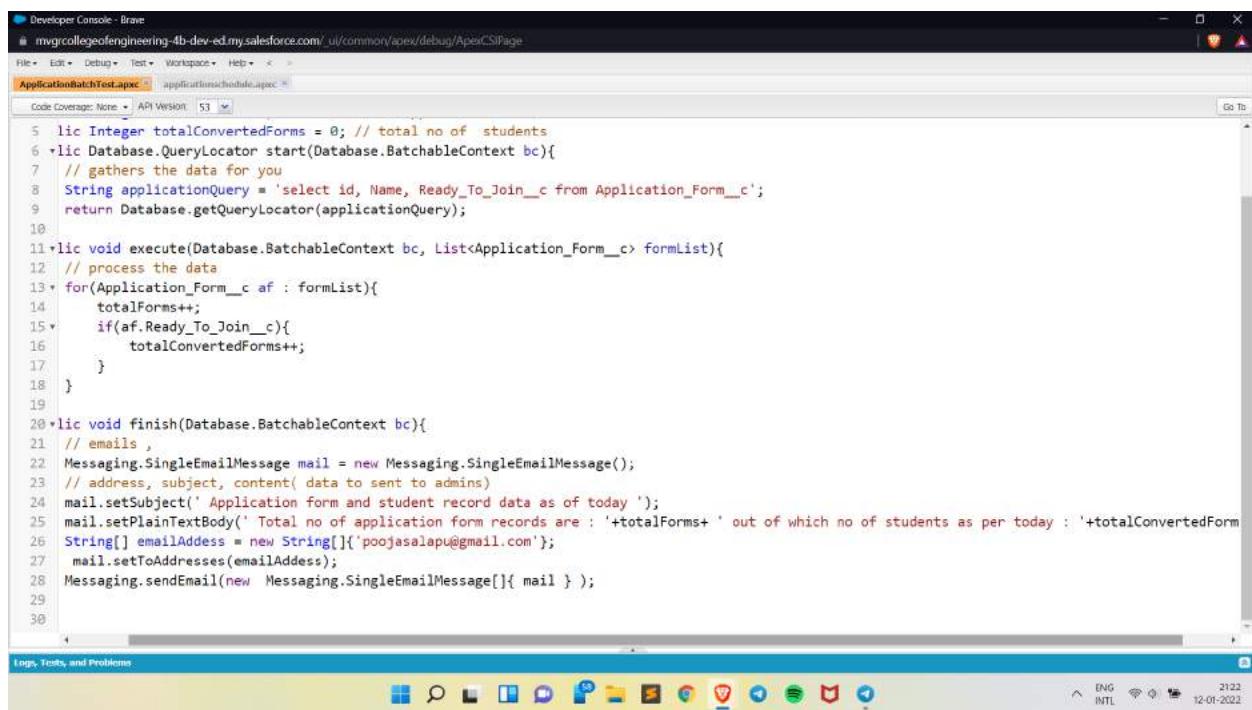
We create ApplicationBatchTest and applicationschedule classes to schedule jobs. From setup search for apex class and click on schedule jobs and fill the details as per your requirements.

# PROJECT REPORT

Upload the Screenshot the Milestone / Activities :

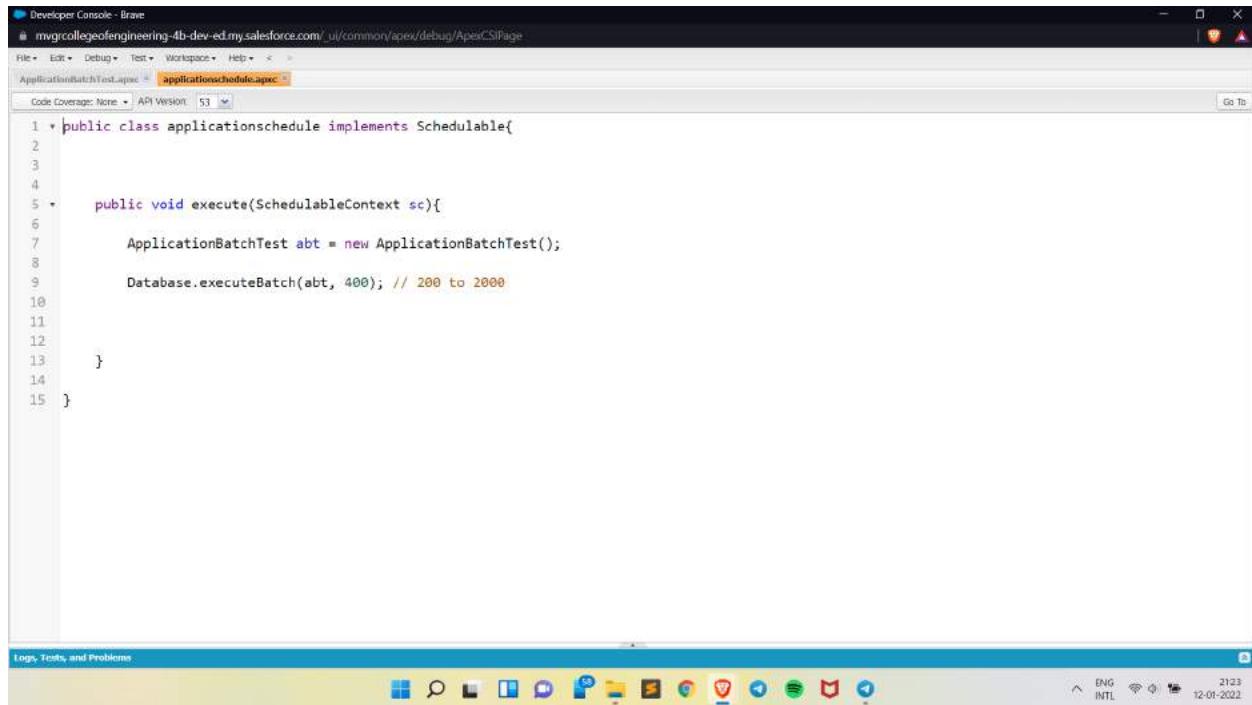


```
1 * public class ApplicationBatchTest implements Database.Batchable<sobject>, Database.stateful {
2
3     //start(), execute(). finish()
4     public Integer totalForms = 0; // total no of application form
5     public Integer totalConvertedForms = 0; // total no of students
6     public Database.QueryLocator 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[] emailAddress = new String[]{poojasalapu@gmail.com};
27
28
29
30 }
```



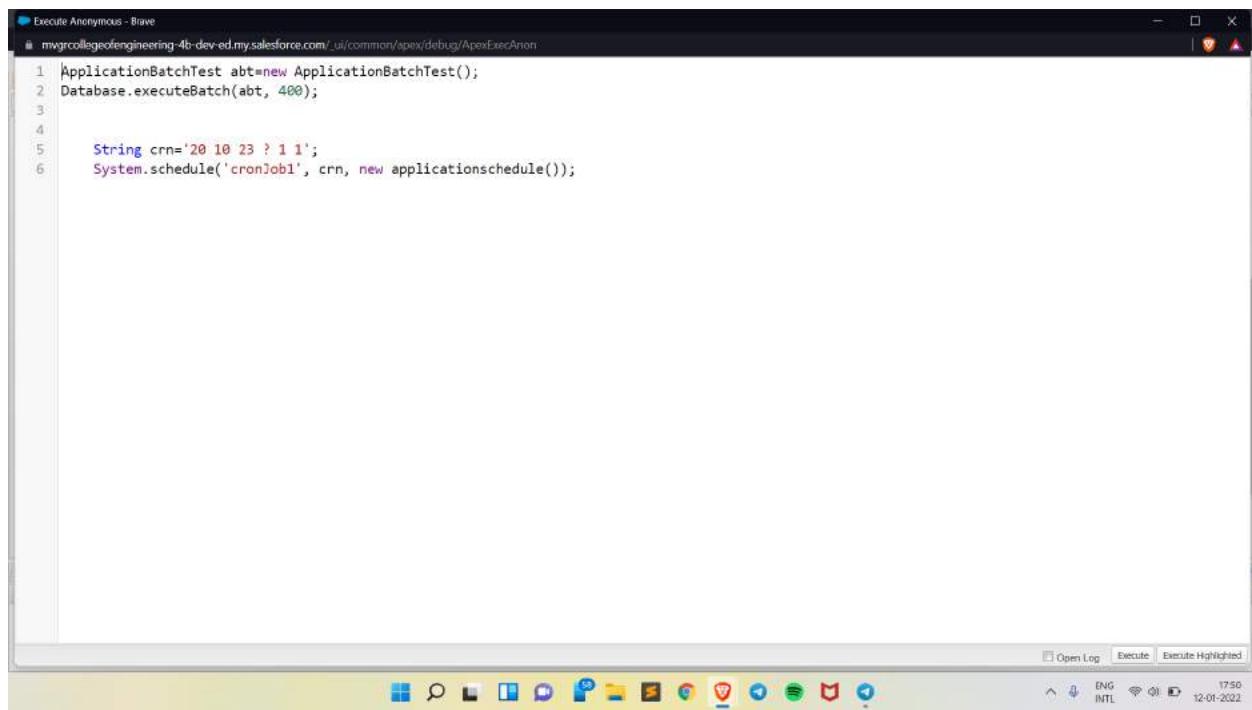
```
5 lic Integer totalConvertedForms = 0; // total no of students
6 *lic Database.QueryLocator start(Database.BatchableContext bc){
7 // gathers the data for you
8 String applicationQuery = 'select id, Name, Ready_To_Join__c from Application_Form__c';
9 return Database.getQueryLocator(applicationQuery);
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 : '+totalConvertedForms+
26 String[] emailAddress = new String[]{poojasalapu@gmail.com};
27 mail.setToAddresses(emailAddress);
28 Messaging.sendEmail(new Messaging.SingleEmailMessage[]{ mail } );
29
30 }
```

# PROJECT REPORT



```
Developer Console - Brave
mvcollegeofengineering-4b-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCIPage
File Edit Debug Test Workspace Help < >
ApplicationBatchTest.apc applicationschedule.apc
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 }
```

The screenshot shows the Salesforce Developer Console in a Brave browser window. The URL is mvcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexCIPage. The tab bar has 'ApplicationBatchTest.apc' and 'applicationschedule.apc' selected. The code editor displays the 'applicationschedule.apc' file, which implements the Schedulable interface. It contains a single execute method that creates an instance of ApplicationBatchTest and executes a batch of 400 records.



```
Execute Anonymous - Brave
mvcollegeofengineering-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 cron='20 10 23 ? 1 1';
6 System.schedule('cronJob1', cron, new applicationschedule());
```

The screenshot shows the Salesforce Developer Console in a Brave browser window. The URL is mvcollegeofengineering-4b-dev-ed.my.salesforce.com/\_ui/common/apex/debug/ApexExecAnon. The code block demonstrates how to create an ApplicationBatchTest instance, execute a batch of 400 records, and schedule a cron job to run at 20 10 23 ? 1 1 using the System.schedule method.

# PROJECT REPORT

**Apex Jobs**

Monitor the status of all Apex jobs, and optionally, abort jobs that are in progress.

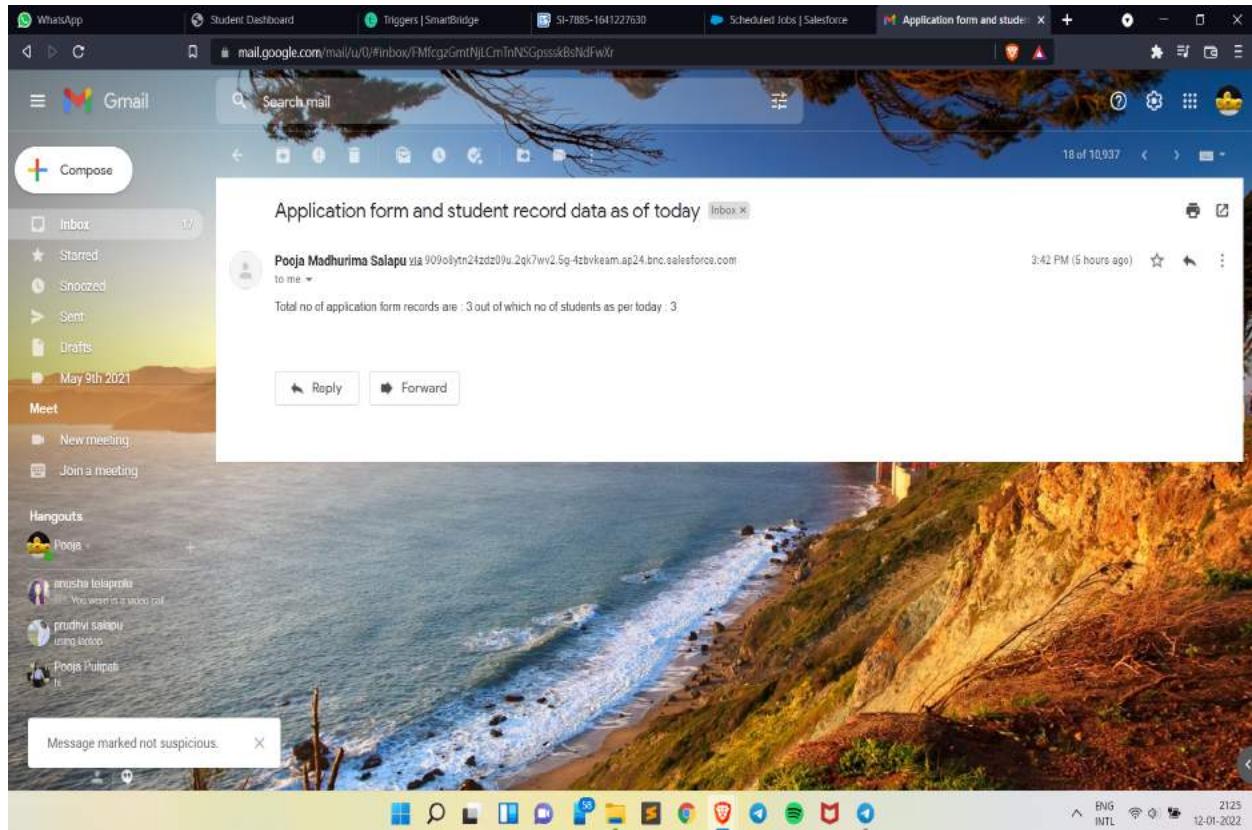
Action	Submitted Date	Job Type	Status	Status Detail	Total Batches	Batches Processed	Failures	Submitted By	Completion Date	Apex Class	Apex Method	Apex Job ID
1/12/2022, 2:30 AM	Batch Apex	Completed			1	1	0	Salapu_Pooja_Machurima	1/12/2022, 2:30 AM	ApplicationBatchTest		7075g00001TdSE
1/12/2022, 2:30 AM	Scheduled Apex	Queued			0	0	0	Salapu_Pooja_Machurima		applicationschedule		7075g00001TdSG
1/12/2022, 2:15 AM	Scheduled Apex	Aborted			0	0	0	Salapu_Pooja_Machurima		applicationschedule		7075g00001TcZB
1/12/2022, 2:12 AM	Batch Apex	Completed			1	1	0	Salapu_Pooja_Machurima	1/12/2022, 2:12 AM	ApplicationBatchTest		7075g00001Tdf6
1/12/2022, 2:09 AM	Batch Apex	Completed		First error: SendEmail failed. First exception on row 0; first error: FIELD_IS_DEFINED: ID is invalid or you do not have access to the record. [toAddresses, your]	1	1	0	Salapu_Pooja_Machurima	1/12/2022, 2:09 AM	ApplicationBatchTest		7075g00001fcbu

**Scheduled Jobs**

The All Scheduled Jobs page lists all of the jobs scheduled by your users. Multiple job types may display on this page. You can delete scheduled jobs if you have the permission to do so.

Action	Job Name	Submitted By	Submitted	Started	Next Scheduled Run	Type
Del	cronJob1	Salapu_Pooja_Machurima	1/12/2022, 2:30 AM		1/10/2022, 11:10 PM	Scheduled Apex
Del	Analytics Data Loader Job for Org 00D5g00004zBVk	User_Integration	6/12/2021, 8:13 PM	1/11/2022, 8:37 PM	1/12/2022, 8:37 PM	Autonomous Data Loader Job

# PROJECT REPORT



Day 7 :

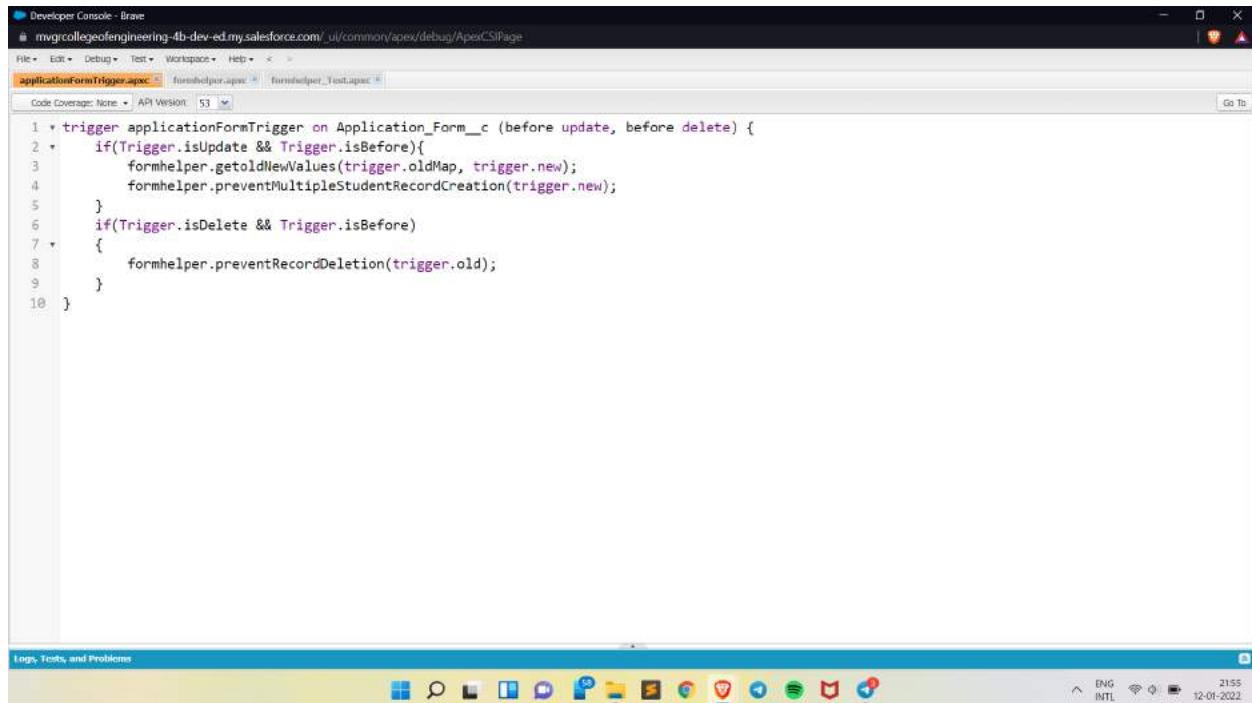
Topic: Triggers.

Milestone / Activities: To know about how triggers are used. Create trigger classes for application form object.

Detailed Description: Open developer console and click on New. Click on the trigger and give it a name. Before insert, delete, update and after insert, delete and update operations are tracked or set by triggers.

# PROJECT REPORT

Upload the Screenshot the Milestone / Activities :



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

File • Edit • Debug • Test • Workspace • Help • < >

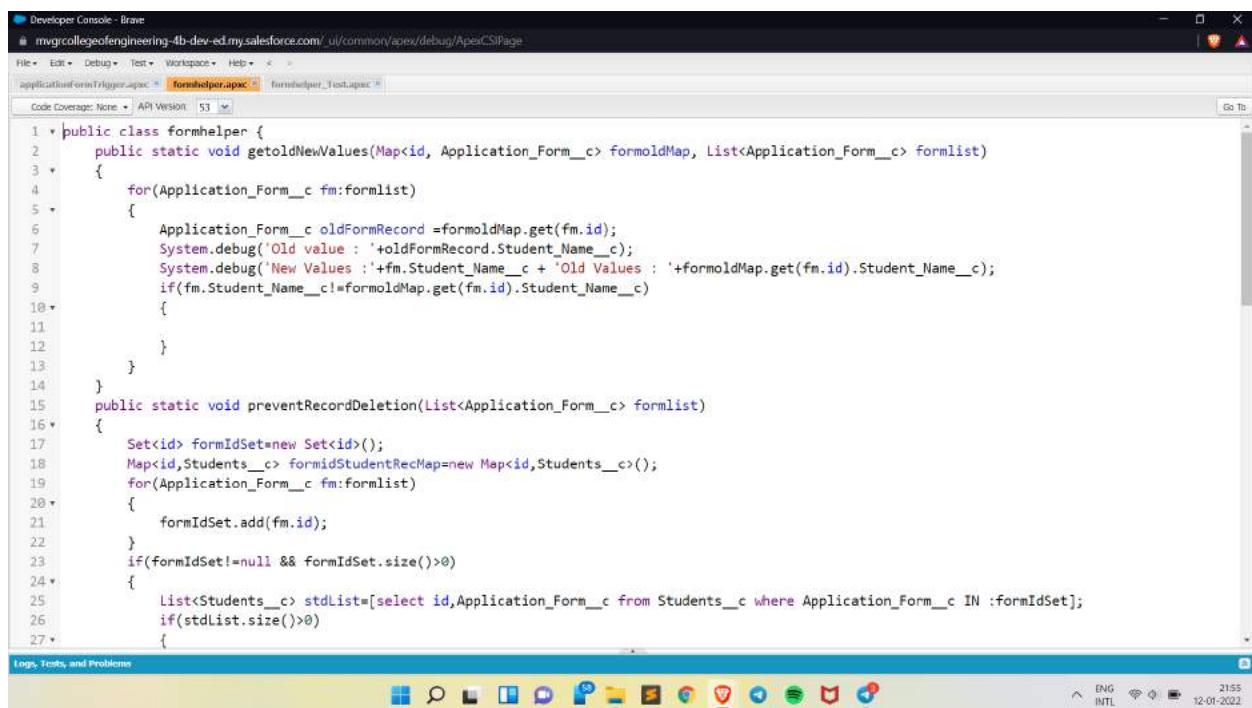
applicationFormTrigger.apc formhelper.apc formhelper\_Test.apc

Code Coverage: None API Version: 53 Go To

```
1 * trigger applicationFormTrigger on Application_Form__c (before update, before delete) {
2 +   if(Trigger.isUpdate && Trigger.isBefore){
3     formhelper.getOldNewValues(trigger.oldMap, trigger.new);
4     formhelper.preventMultipleStudentRecordCreation(trigger.new);
5   }
6   if(Trigger.isDelete && Trigger.isBefore)
7   {
8     formhelper.preventRecordDeletion(trigger.old);
9   }
10 }
```

Logs, Tests, and Problems

Windows Taskbar: ENG INTL 2155 12-01-2022



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

File • Edit • Debug • Test • Workspace • Help • < >

applicationFormTrigger.apc formhelper.apc formhelper\_Test.apc

Code Coverage: None API Version: 53 Go To

```
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 }
15 public static void preventRecordDeletion(List<Application_Form__c> formlist)
16 {
17   Set<id> formIdSet=new Set<id>();
18   Map<id,Students__c> formidStudentRecMap=new Map<id,Students__c>();
19   for(Application_Form__c fm:formlist)
20   {
21     formIdSet.add(fm.id);
22   }
23   if(formIdSet!=null && formIdSet.size()>0)
24   {
25     List<Students__c> stdList=[select id,Application_Form__c from Students__c where Application_Form__c IN :formIdSet];
26     if(stdList.size()>0)
27     {
28     }
```

Logs, Tests, and Problems

Windows Taskbar: ENG INTL 2155 12-01-2022

# PROJECT REPORT

The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Brave" and the URL is "mvrcollegeofengineering-4b-dev-ed.my.salesforce.com/ui/common/apex/debug/ApexCSVPage". The tabs at the top are "applicationFormTrigger.apex", "formhelper.apex", and "formhelper\_Test.apex". The main area displays the following Apex code:

```
27 *
28     {
29         for(Students__c std:stdList){
30             formidstudentRecMap.put(std.Application_Form__c,std);
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)
```

The status bar at the bottom right shows "2155 ENG INTL 12-01-2022".

The screenshot shows the Salesforce Developer Console interface, identical to the first one but with some changes in the code. The title bar and URL are the same. The tabs are "applicationFormTrigger.apex", "formhelper.apex", and "formhelper\_Test.apex". The main area displays the following modified Apex code:

```
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)
54         {
55             for(Students__c std:stdList){
56                 formidStudentRecMap.put(std.Application_Form__c,std);
57
58                 Subjects__c sub=new Subjects__c();
59                 sub.Student__c = std.id;
60                 stdlist2.add(sub);
61             }
62         }
63         if(stdlist2.size()>0)
64         {
65             insert stdlist2;
66         }
67         if(formidStudentRecMap!=null && formidStudentRecMap.values().size()>0)
68     }
```

The status bar at the bottom right shows "2156 ENG INTL 12-01-2022".

# PROJECT REPORT

The screenshot shows the Salesforce Developer Console interface. The top bar includes 'Developer Console - Brave' and the URL 'mvrcollegeofengineering-4b-dev-ed.my.salesforce.com/.ui/common/apex/debug/ApexCSVPage'. The menu bar has options like File, Edit, Debug, Test, Workspace, Help, and a Go To button. The code editor tab is 'formhelper.apc'. The code itself is an Apex trigger named 'applicationFormTrigger.apc'.

```
52 List<Students__c> stdList=[select id, Application_Form__c from Students__c where Application_Form__c IN :formIdSet];
53 if(stdList.size()>0)
54 {
55     for(Students__c std:stdList){
56         formidStudentRecMap.put(std.Application_Form__c,std);
57
58         Subjects__c sub=new Subjects__c();
59         sub.Student__c = std.id;
60         stdlist2.add(sub);
61     }
62 }
63 if(stdlist2.size()>0)
64 {
65     insert stdlist2;
66 }
67 if(formidStudentRecMap!=null && formidStudentRecMap.values().size()>0)
68 {
69     for(Application_Form__c fm:formlist){
70         if(formidStudentRecMap.containsKey(fm.id) && formidStudentRecMap.get(fm.id)!=null){
71             fm.addError('No further changes allowed now');
72         }
73     }
74 }
75 }
76 }
77 }
78 }
```

The status bar at the bottom shows 'Logs, Tests, and Problems', the system tray with icons for ENG, INTL, and battery level, and the date/time '2156 12-01-2022'.

The screenshot shows the Salesforce Developer Console interface. The top bar includes 'Developer Console - Brave' and the URL 'mvrcollegeofengineering-4b-dev-ed.my.salesforce.com/.ui/common/apex/debug/ApexCSVPage'. The menu bar has options like File, Edit, Debug, Test, Workspace, Help, and a Run Test/Go To button. The code editor tab is 'formhelper\_Test.apc'. The code is a test class named 'formhelper\_Test'.

```
2 * public class formhelper_Test {
3 *     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='herry@herry.com';
15    af.Student_Name__c='Herry herry';
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='Herry 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);
```

The status bar at the bottom shows 'Logs, Tests, and Problems', the system tray with icons for ENG, INTL, and battery level, and the date/time '2156 12-01-2022'.

# PROJECT REPORT

Day 8 :

Topic: Javascript, VS Code, Salesforce CLI, LWC Setup.

Milestone / Activities: Javascript introduction, VS code install and salesforce cli setup. Salesforce LWC setup and working from VS code to connect to org. To deploy and retrieve the codes from LWC, VS Code to org.

Detailed Description: Javascript - Can be done in chrome console by right click and selecting inspect option.

Download VS Code and download all required project manager and java and salesforce extensions. SFDX is compulsory. Click Ctrl+Shift+P to open the command palette and click create project with manifest and click standard. Now create a project folder called 'collegeManagement'. Next to connect to our org. Click authorize org. It will be redirected to the login page. Validate and get in to org.

Changes made in code in VS code can be reflected in org by clicking the deploy to org option and can be retrieved from org by clicking retrieve from source org. Click on the LWC extension and an beta version opens. Here we can create lightning components and can be deployed to the org by deploying it in LWC beta software.

Upload the Screenshot the Milestone / Activities :

```
PS C:\Users\pooya>sf dx
Microsoft Windows [Version 10.0.22000.434]
(c) Microsoft Corporation. All rights reserved.

C:\Users\pooya>sf dx
salesforce CLI

VERSION
  sf dx-cll/7.133.0 win32-x64 node-v16.13.1

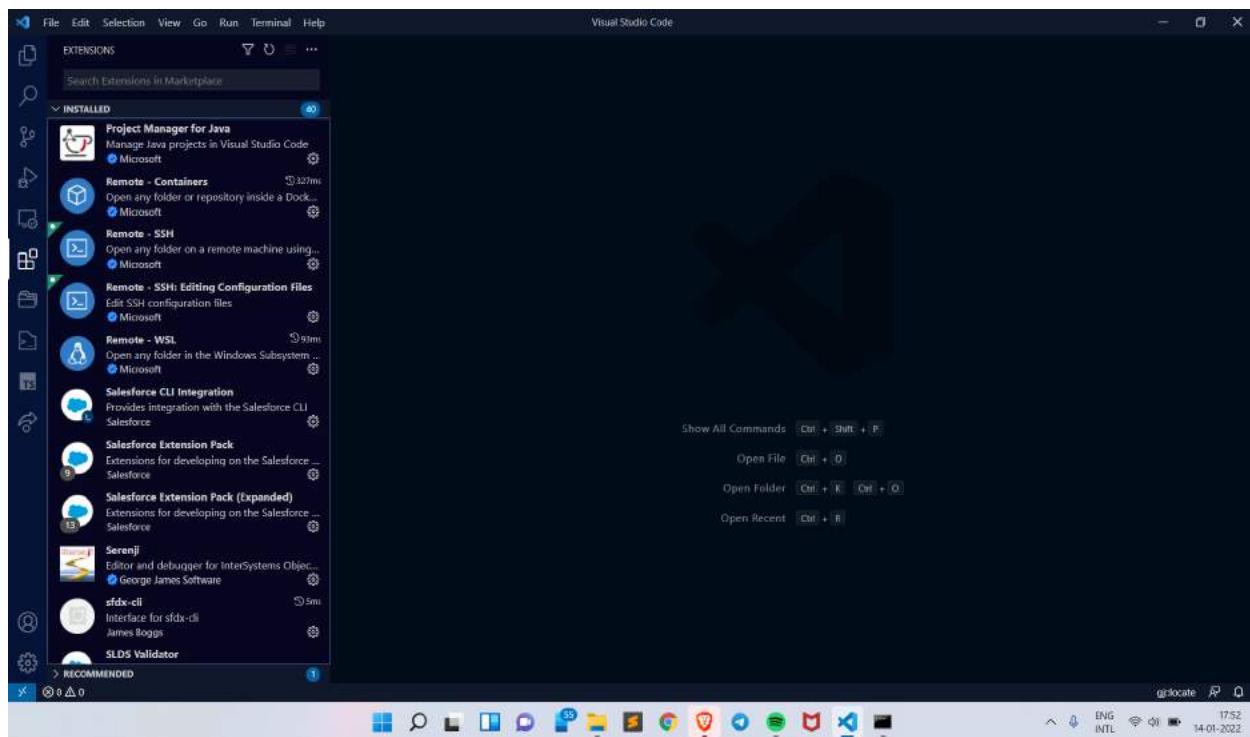
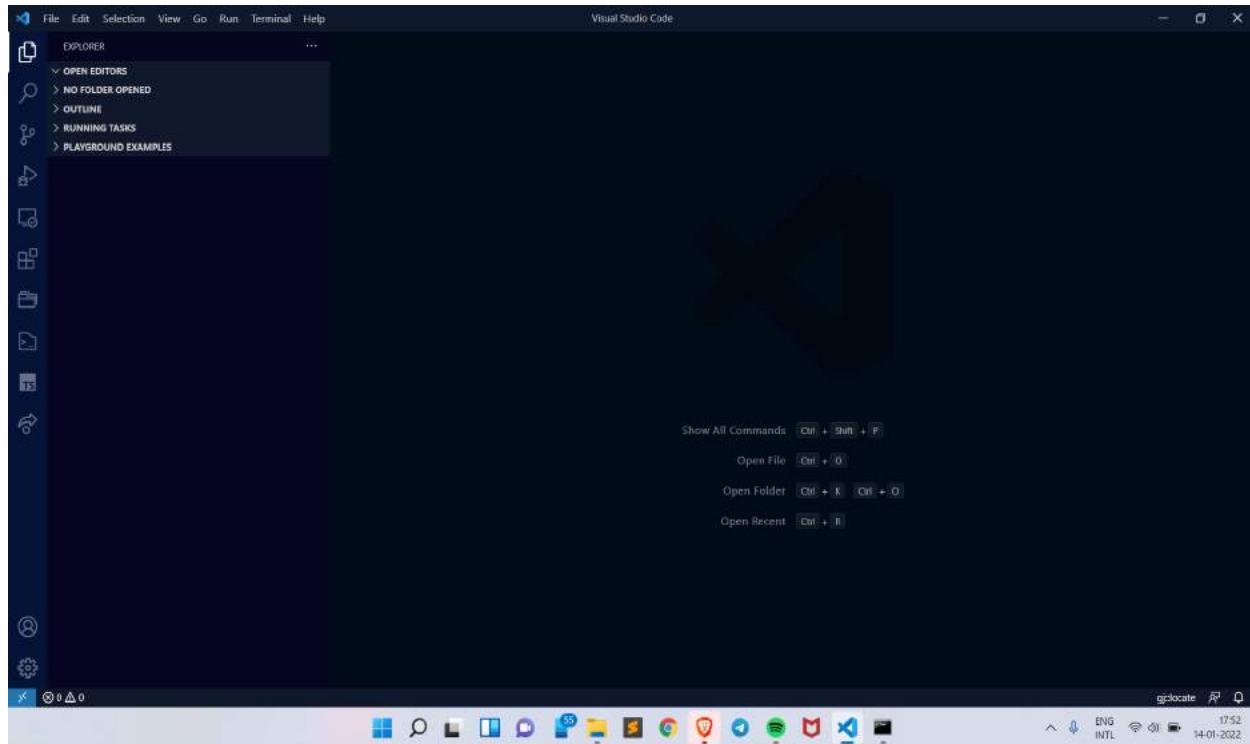
USAGE
  $ sf dx [COMMAND]

TOPICS
  alias      manage username aliases
  auth       authorize an org for use with the Salesforce CLI
  config    configure the Salesforce CLI
  force     tool for Salesforce developers
  help      access CLI info from the command line
  plugin   add/remove/create CLI plug-ins

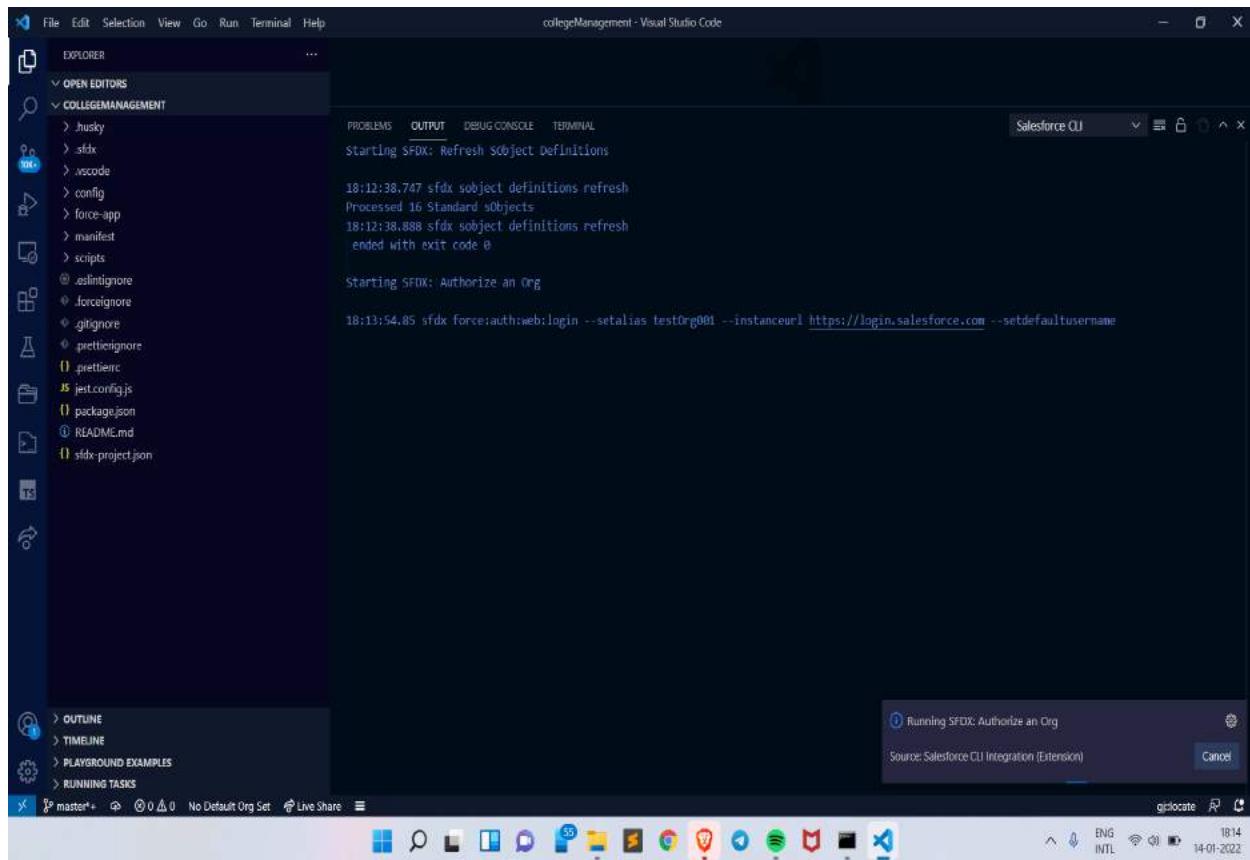
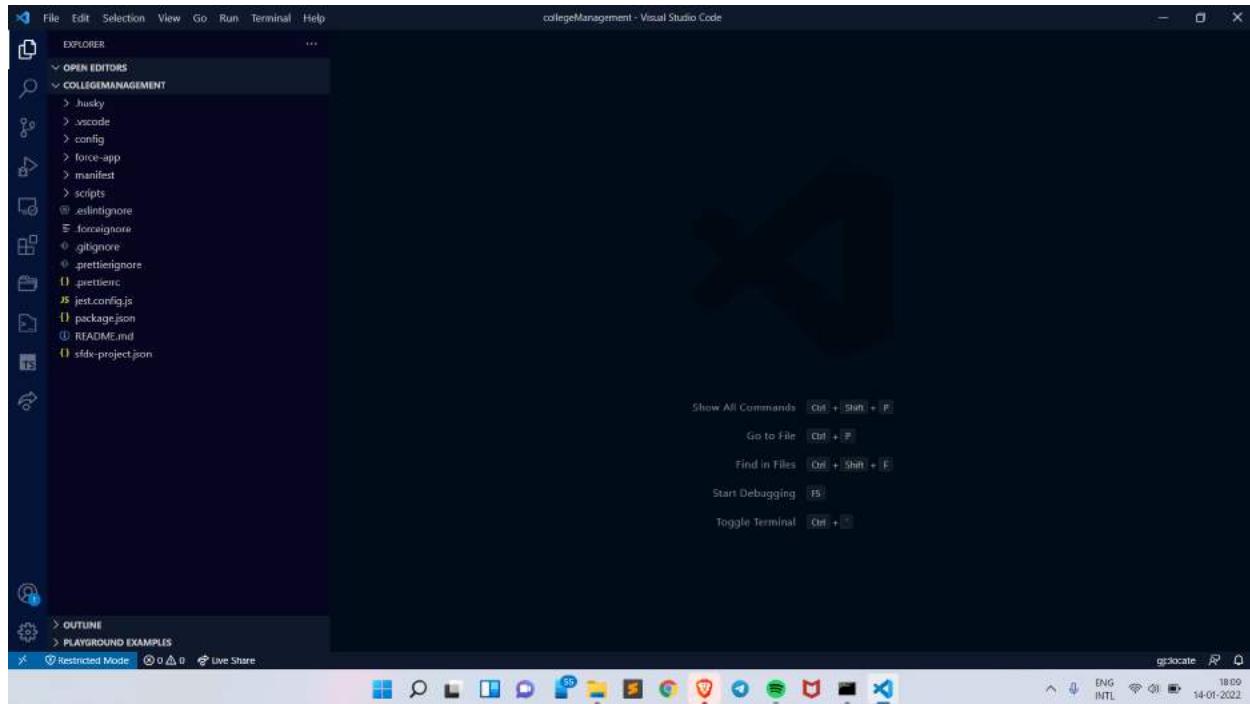
COMMANDS
  autocomplete  display autocomplete installation instructions
  commands     list all the commands
  help         display help for sf dx
  plugin      list installed plugins
  update      update sf dx
  which       show which plugin a command is in

C:\Users\pooya>
```

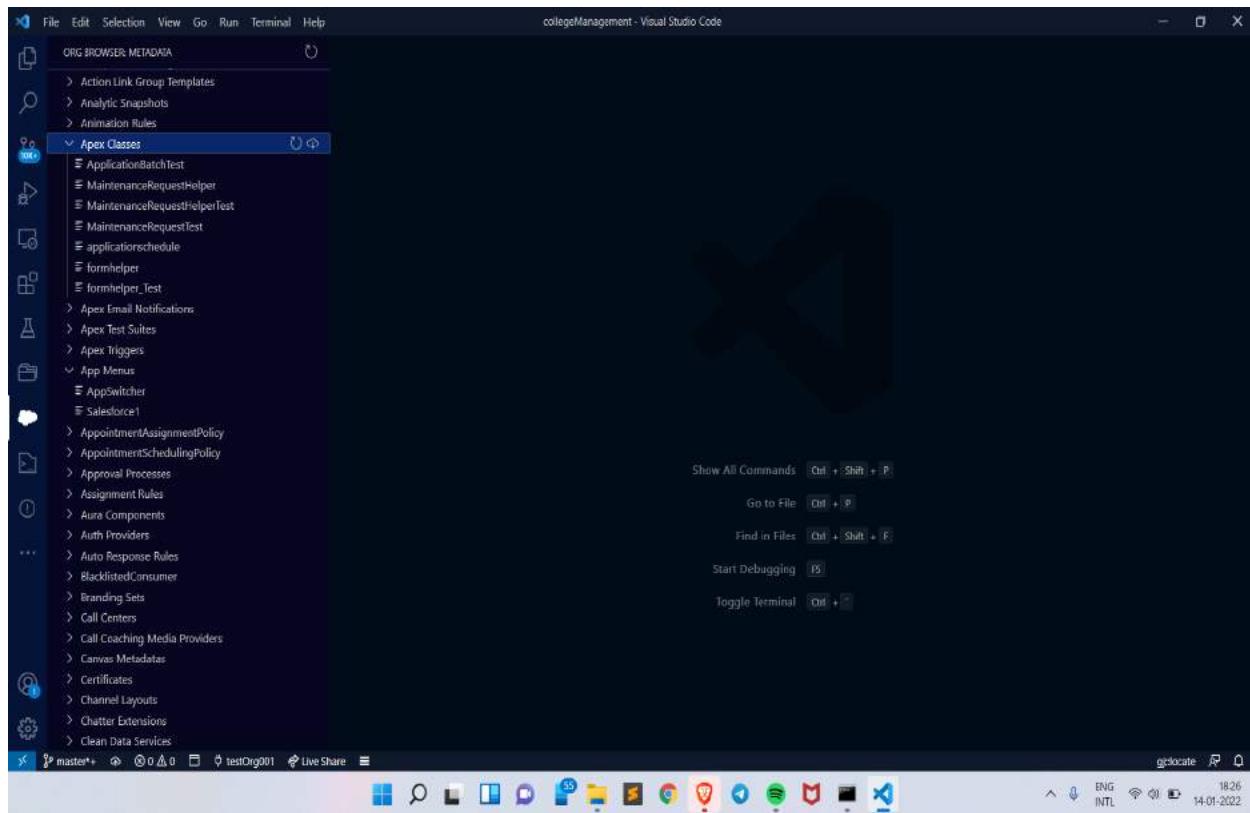
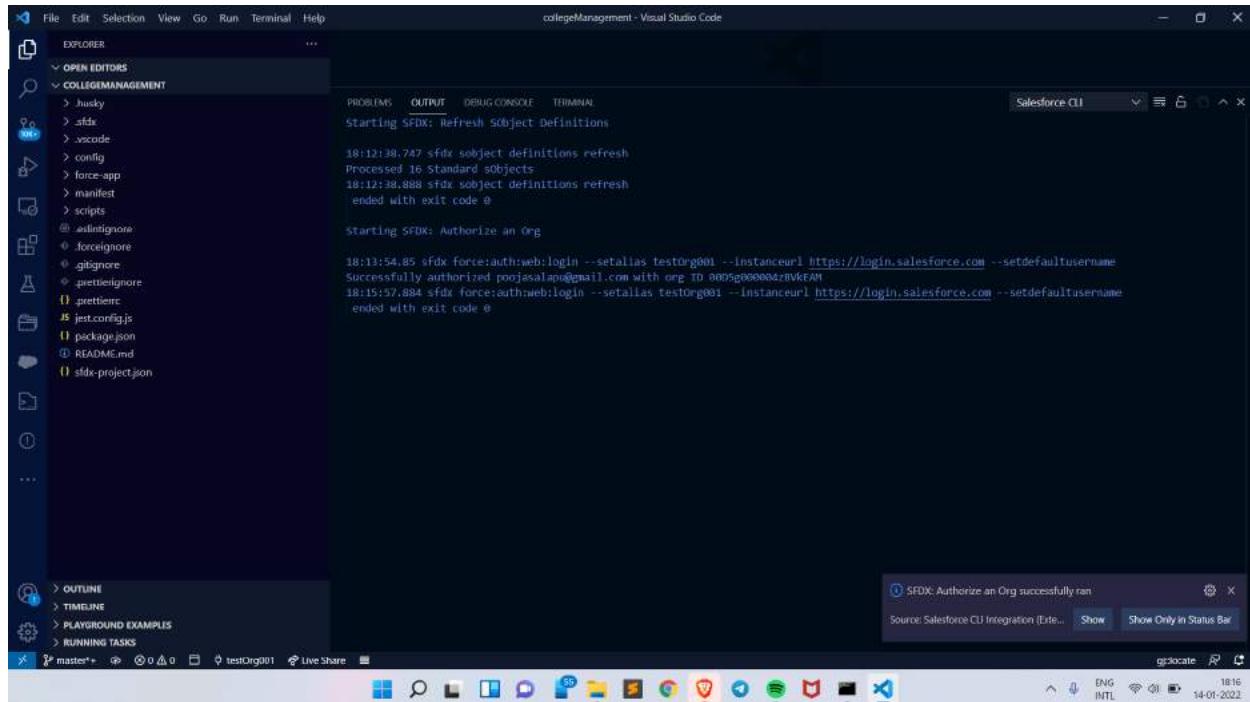
# PROJECT REPORT



# PROJECT REPORT



# PROJECT REPORT



# PROJECT REPORT

The screenshot shows the Visual Studio Code interface with the "collegeManagement" project open. The left sidebar displays the "ORG BROWSER: METADATA" tree, which includes sections for Action Link Group Templates, Analytic Snapshots, Animation Rules, Apex Classes, Apex Email Notifications, Apex Test Suites, Apex Triggers, App Menus, AppSwitcher, and various Salesforce components like AppointmentAssignmentPolicy, AppointmentSchedulingPolicy, Assignment Rules, Aura Components, Auth Providers, Auto Response Rules, BlacklistedConsumer, Branding Sets, Call Centers, Call Coaching Media Providers, Canvas Metadatas, Certificates, Channel Layouts, Chatter Extensions, and Clean Data Services.

The main editor area shows the output of a Salesforce CLI command:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
18:24:16,819 sfdx force:auth:web:login --setalias testOrg001 --instanceurl https://login.salesforce.com --setdefaultusername
ended with exit code 0

Starting SFDX: Set a Default Org
18:25:05,782 sfdx force:config:set defaultusername=testOrg001
--- Set Config
Name Value Success
defaultusername testOrg001 true
18:25:08,275 sfdx force:config:set defaultusername=testOrg001
ended with exit code 0

18:27:23,110 Starting SFDX: Retrieve Source from Org
--- Retrieved Source
FULL NAME TYPE PROJECT PATH
applicationschedule ApexClass force-app/main/default/classes/applicationschedule.cls
applicationschedule ApexClass force-app/main/default/classes/applicationschedule.cls-meta.xml
MaintenanceRequestHelper ApexClass force-app/main/default/classes/MaintenanceRequestHelper.cls
force-app/main/default/classes/MaintenanceRequestHelper.cls-meta.xml
MaintenanceRequestTest ApexClass force-app/main/default/classes/MaintenanceRequestTest.cls
force-app/main/default/classes/MaintenanceRequestTest.cls-meta.xml
MaintenanceRequestTest ApexClass force-app/main/default/classes/MaintenanceRequestTest.cls
force-app/main/default/classes/MaintenanceRequestTest.cls-meta.xml
formhelper ApexClass force-app/main/default/classes/formhelper.cls
force-app/main/default/classes/formhelper.cls-meta.xml
formhelper ApexClass force-app/main/default/classes/formhelper_test.cls
force-app/main/default/classes/formhelper_test.cls-meta.xml
ApplicationBatchTest ApexClass force-app/main/default/classes/ApplicationBatchTest.cls
force-app/main/default/classes/ApplicationBatchTest.cls-meta.xml
formhelper ApexClass force-app/main/default/classes/formhelper.cls
force-app/main/default/classes/formhelper.cls-meta.xml
MaintenanceRequestHelperTest ApexClass force-app/main/default/classes/MaintenanceRequestHelperTest.cls
force-app/main/default/classes/MaintenanceRequestHelperTest.cls-meta.xml
MaintenanceRequestHelperTest ApexClass force-app/main/default/classes/MaintenanceRequestHelperTest.cls
force-app/main/default/classes/MaintenanceRequestHelperTest.cls-meta.xml
18:27:24,797 ended SFDX: Retrieve Source from Org
```

The status bar at the bottom indicates the file is "master" and the org is "testOrg001".

The screenshot shows the Visual Studio Code interface with the "myinstLWC.html" file open in the editor. The file contains the following LWC template code:

```
<template>
<h1> I am learning LWC </h1>
</template>
```

The left sidebar shows the project structure under "COLLEGEMANAGEMENT" with files like myInstLWC.html, myInstLWC.js, myInstLWC.js-meta.xml, .eslintrc.json, .jshintrc.json, account.sobj, .eslintignore, .forceignore, .gitignore, .prettierignore, hello.apex, and account.sobj.

A status bar at the bottom right shows a deployment task: "Running SFDX: Deploy Source to Org" with a progress bar and a "Cancel" button. The status bar also indicates the source is "Salesforce CLI Integration (Extension)".

# PROJECT REPORT

The screenshot shows the Salesforce Setup interface for managing Lightning Components. The left sidebar includes links for Setup Home, Service Setup Assistant, Multi-Factor Authentication Assistant, Release Updates, Lightning Experience Transition Assistant, New Salesforce Mobile App QuickStart, Lightning Usage, Optimizer, Administration (Users, Data, Email), and Platform Tools (Apps, Feature Settings). The main content area displays the "Lightning Components" page with a heading "Lightning Components". It includes a brief description of what a Lightning component is and how it can be used. Below this is a table listing components, with one entry: "myinstLWC" (Label: myinstLWC, Type: LWC). The table has columns for Action, Name, Label, Type, Namespace Prefix, and API Version (52.0). A "View" dropdown menu is at the top of the table.

The screenshot shows the Salesforce LWC Editor (Beta) interface. The left sidebar lists "APEX CLASSES" and "LIGHTNING WEB COMPONENTS" under "myinstLWC". The main area is titled "Welcome" and contains the text: "The Salesforce LWC Editor is the code editor that powers VS Code.". It features a code editor window with "Component.html" selected, displaying the following LWC component code:

```
<template>
    <div>Hello, world!</div>
</template>

<script>
    import { LightningElement } from 'lwc';
    import { track } from 'lwc';
    import { wireCallout, wireGetRecord } from 'lightning/uiRecordApi';

    export class MyInstLwc extends LightningElement {
        @track recordId;
        @track error;
        @track isCalloutOpen = false;
        @track isRecordLoaded = false;

        @wire(wireGetRecord, { recordId: '$recordId' })
        handleRecord(record) {
            if (record) {
                this.recordId = record.id;
                this.error = undefined;
                this.isRecordLoaded = true;
            }
        }

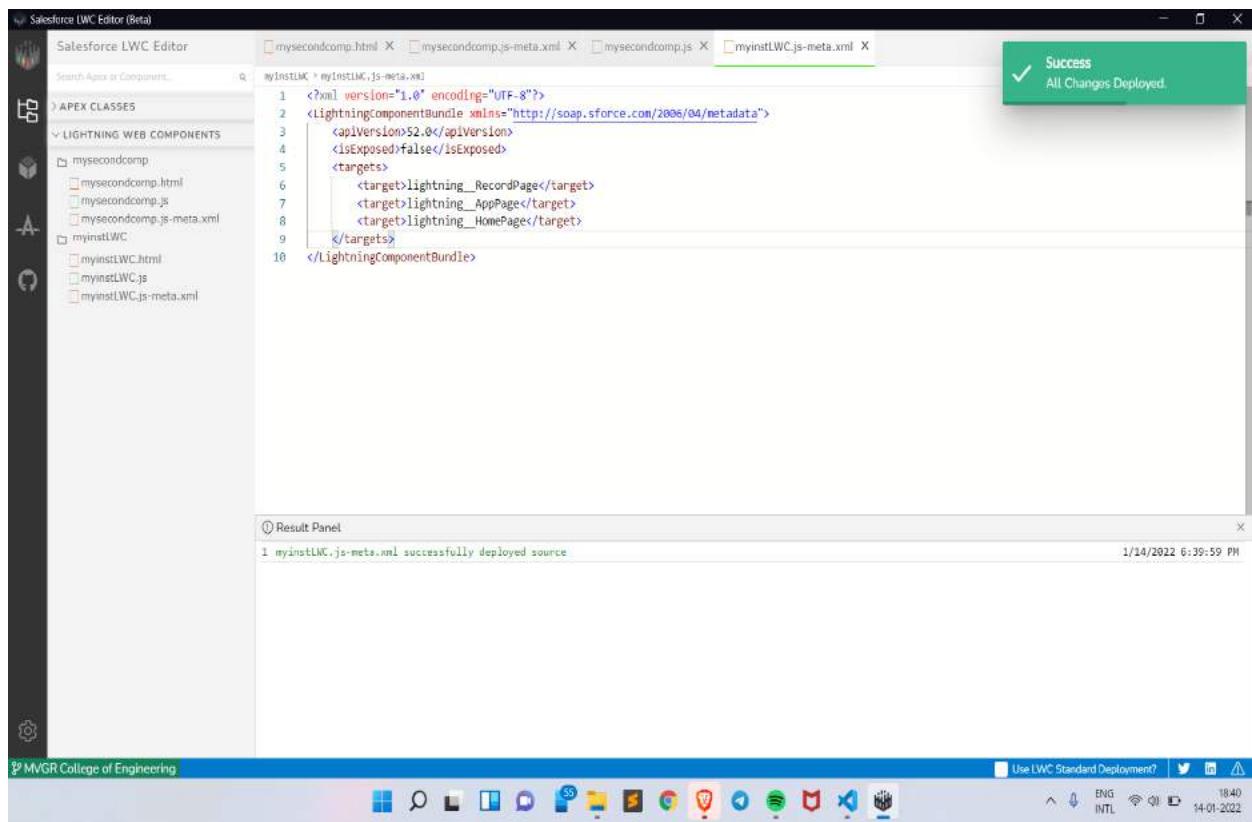
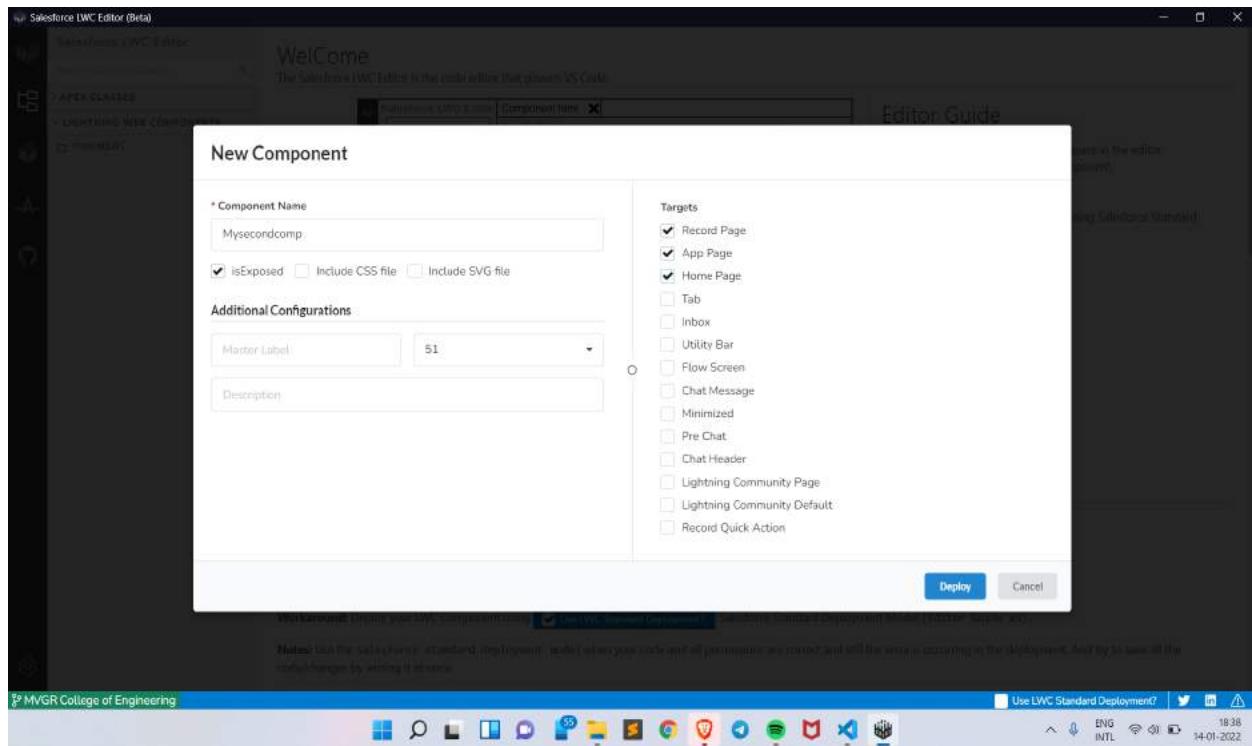
        @wire(wireCallout)
        handleCallout(result) {
            if (result.error) {
                this.error = result.error;
            } else {
                this.isCalloutOpen = true;
            }
        }
    }
</script>
```

The editor interface includes a sidebar with icons for file operations, a search bar, and a "Component.html" tab. Numbered arrows (1-6) point to specific UI elements: 1 points to the sidebar icon, 2 points to the "APEX CLASSES" section, 3 points to the "LIGHTNING WEB COMPONENT" section, 4 points to the component code, 5 points to the "Use LWC Standard Deployment?" checkbox, and 6 points to the deployment status indicator. To the right of the editor is an "Editor Guide" section with numbered steps:

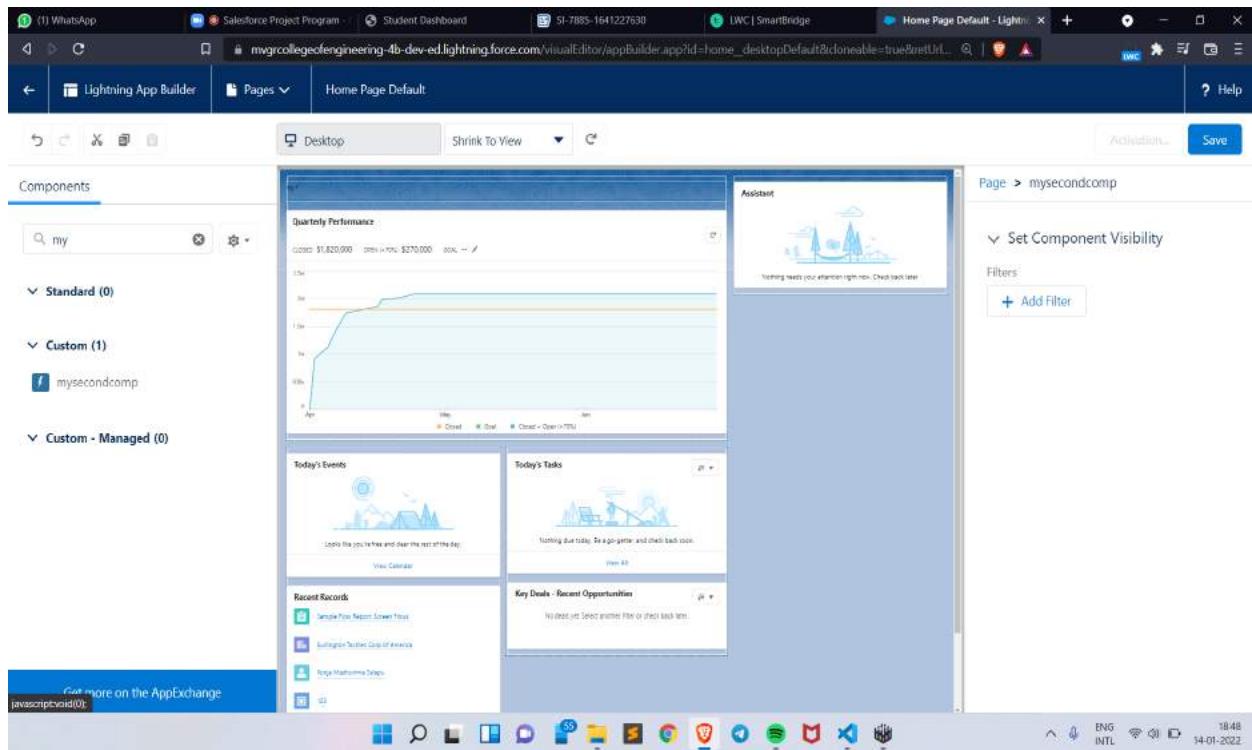
1. Show/hide the sidebar to get more space in the editor.
2. Create new (LWC) lightning web component.
3. Create new apex class.
4. Editor additional settings
5. Deploy Lightning Web Component using Salesforce Standard Deployment (Change Set) Model
6. Deployment errors logs.

Below the editor, there is a "Fixed" section with notes about deployment issues and workarounds. The bottom of the screen shows the Windows taskbar with various pinned icons and the date/time (14-01-2022).

# PROJECT REPORT



# PROJECT REPORT



Day 9 :

Topic: Javascript functions, Lightning web components

Milestone / Activities: To create lightning web components and update from vs code to org.

Detailed Description:

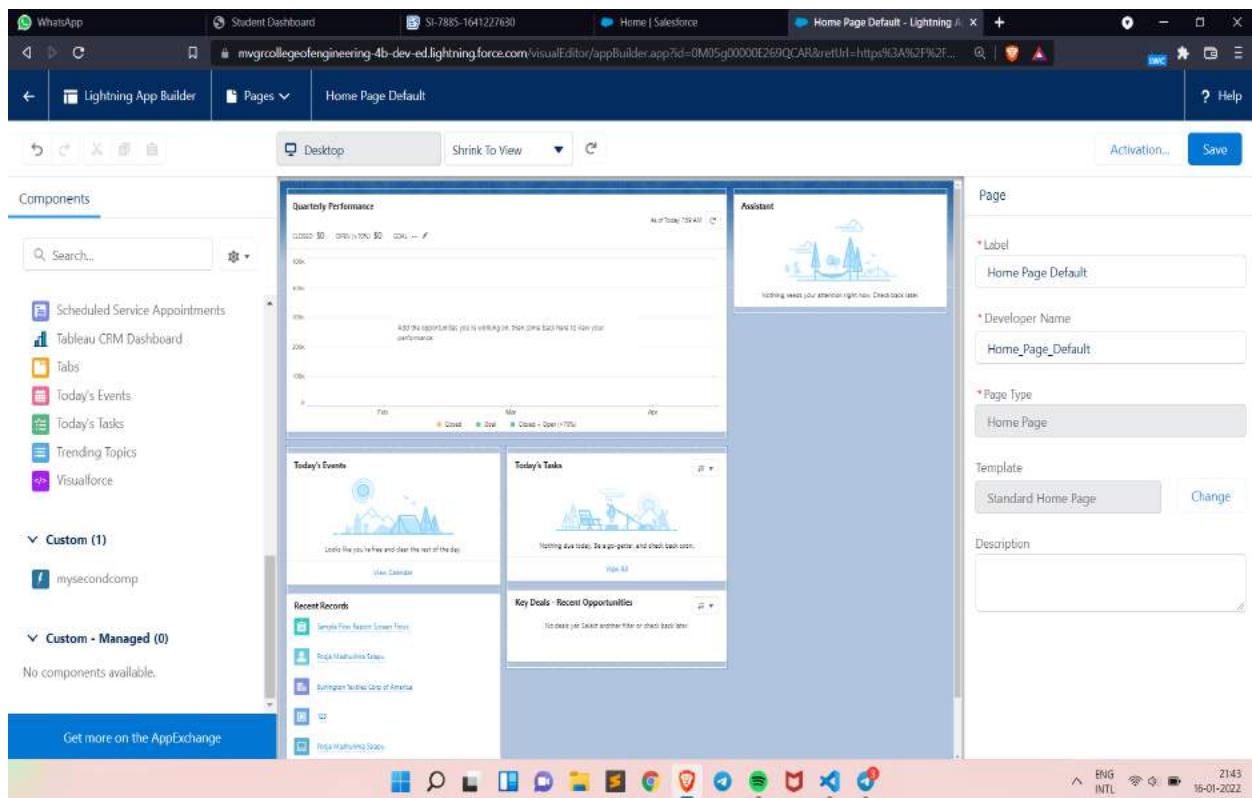
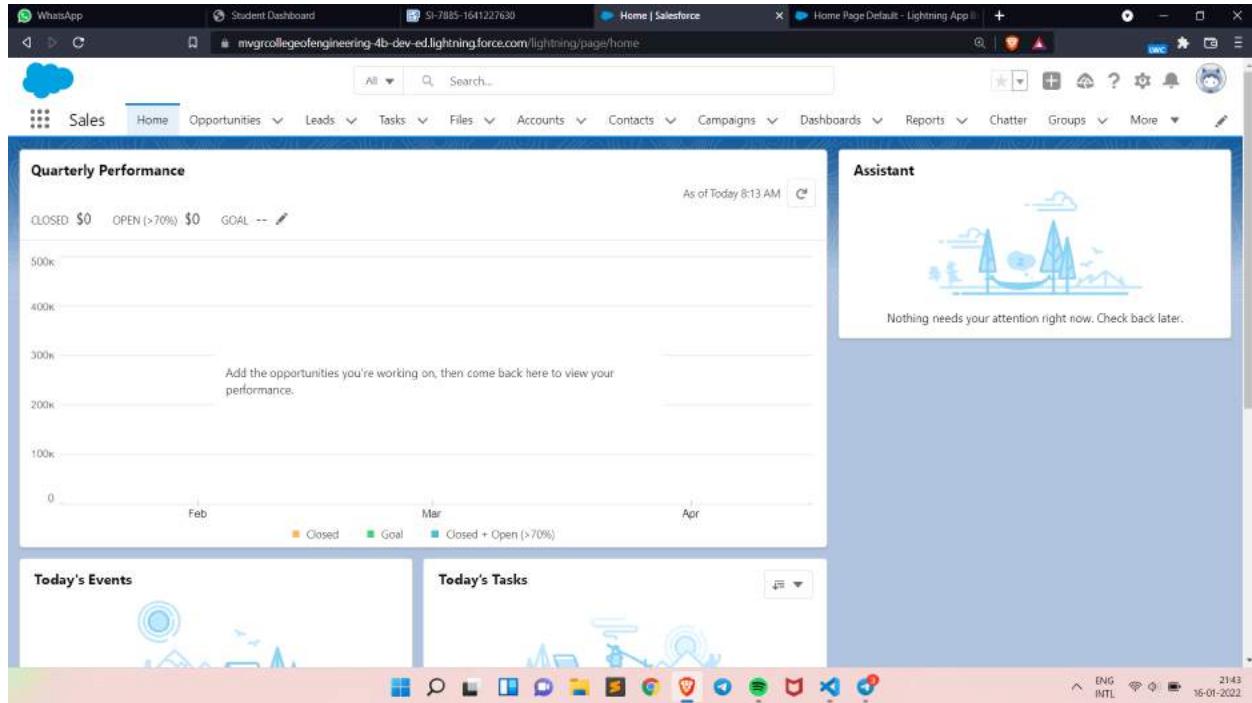
Open vs code and create a lightning web component and deploy them to the org. In the setup page of salesforce, go to sales and click on home tab. All the changes can be seen when the components created are added in lightning web builder.

To add the component, drag the component to the area in the lightning builder page and activate it. Then the changes can be seen in the home tab.

In the code, we can add html,js, and css pages. All the properties of html,css and js can be used. Javascript functions are also used usual. The meta file will contain version and <isExposed> tag as true to be able to make the component visible in the workspace.

# PROJECT REPORT

Upload the Screenshot the Milestone / Activities :



# PROJECT REPORT

File Edit Selection View Go Run Terminal Help mysecondcomp.html - collegeManagement - Visual Studio Code

OPEN EDITORS COLLEGE MANAGEMENT

```
<div><div style="background-color: white;">
    lightning-card title="My lightning card" icon-name="standard:catalog">
        Click the checkbox :
        lightning-input type="checkbox" label="Click me" name="input1" onclick="handleClick()"/>/lightning-input>

        <template if:true={showPersonalInfo}>
            My FullName : {firstname} {lastname}<br/>
            My Email : {email} <br/>
            My Age : {age}
        </template>
</div>
```

force-app > main > default > lwc > mysecondcomp > mysecondcomp.html > template > div > lightning-card

EXPLORER OUTLINE TIMELINE PLAYGROUND EXAMPLES RUNNING TASKS

gitlocate In 13, Col 26 Spaces: 4 UTF-8 LF HTML □ Prettier

master\* testOrg001 Live Share

2143 ENG INTL 16-01-2022

This screenshot shows the Visual Studio Code interface with the 'mysecondcomp.html' file open. The code defines a Lightning component with a card titled 'My lightning card'. It contains a checkbox labeled 'Click me' and a template section for personal information. The file is part of a Salesforce project structure under 'force-app/main/default/lwc/mysecondcomp'.

File Edit Selection View Go Run Terminal Help mysecondcomp.css - collegeManagement - Visual Studio Code

OPEN EDITORS COLLEGE MANAGEMENT

```
color: red;
font-size: 30px;
font-variant: larger;
```

force-app > main > default > lwc > mysecondcomp > mysecondcomp.css > p

EXPLORER OUTLINE TIMELINE PLAYGROUND EXAMPLES RUNNING TASKS

gitlocate In 5, Col 2 Spaces: 4 UTF-8 CRLF CSS □ Prettier

master\* testOrg001 Live Share

2144 ENG INTL 16-01-2022

This screenshot shows the Visual Studio Code interface with the 'mysecondcomp.css' file open. The code contains a single CSS rule for a paragraph element, setting the color to red, font size to 30px, and font variant to larger. The file is part of the same Salesforce project structure as the previous screenshot.

# PROJECT REPORT

Visual Studio Code interface showing the 'mysecondcomp.js' file content. The code defines a class 'Myinstlw' extending 'LightningElement'. It includes properties for fname, lname, age, and email, and a handleClick event handler.

```
force-app > main > default > lwc > mysecondcomp > mysecondcomp.js > Myinstlw.js
1 import { LightningElement, api, track } from 'lwc';
2
3 export default class Myinstlw extends LightningElement {
4     fname = 'Pooja Modhuria';
5     lname = 'Salapu';
6     age = 21;
7     email = 'posjasalapu@gmail.com';
8
9     showPersonalInfo = false;
10    handleClick(event) {
11        this.showPersonalInfo = event.target.checked;
12    }
13}
```

Visual Studio Code interface showing the 'mysecondcomp.js-meta.xml' file content. The XML defines a LightningComponentBundle with an exposed component and three targets: RecordPage, AppPage, and HomePage.

```
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
    <apiVersion>52.0</apiVersion>
    <isExposed>true</isExposed>
    <targets>
        <target>lightning_RecordPage</target>
        <target>lightning_AppPage</target>
        <target>lightning_HomePage</target>
    </targets>
</LightningComponentBundle>
```

# PROJECT REPORT

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure under "COLLEGEMANAGEMENT".
- Editors:** Three files are open: "mysecondcomp.html", "mysecondcomp.css", and "mysecondcomp.js".
- Terminal:** Displays deployment logs:

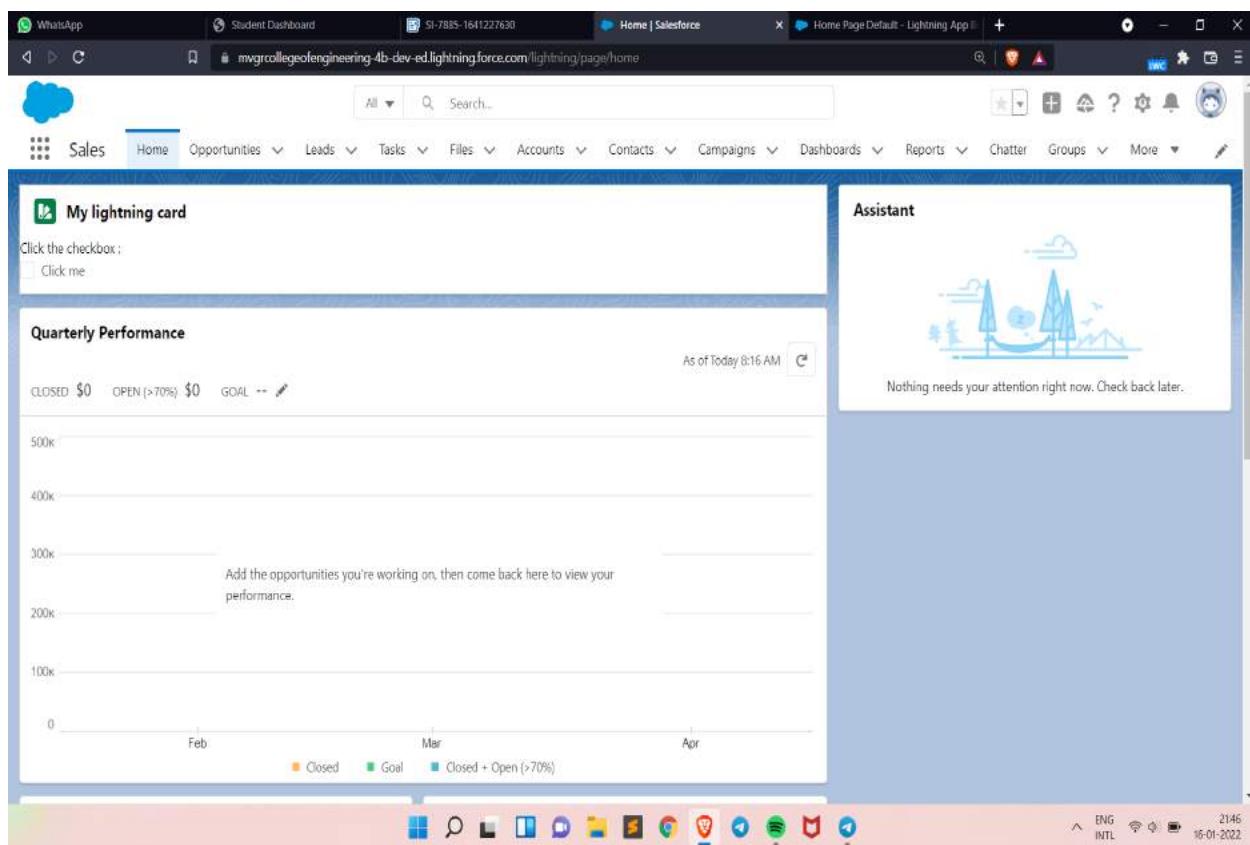
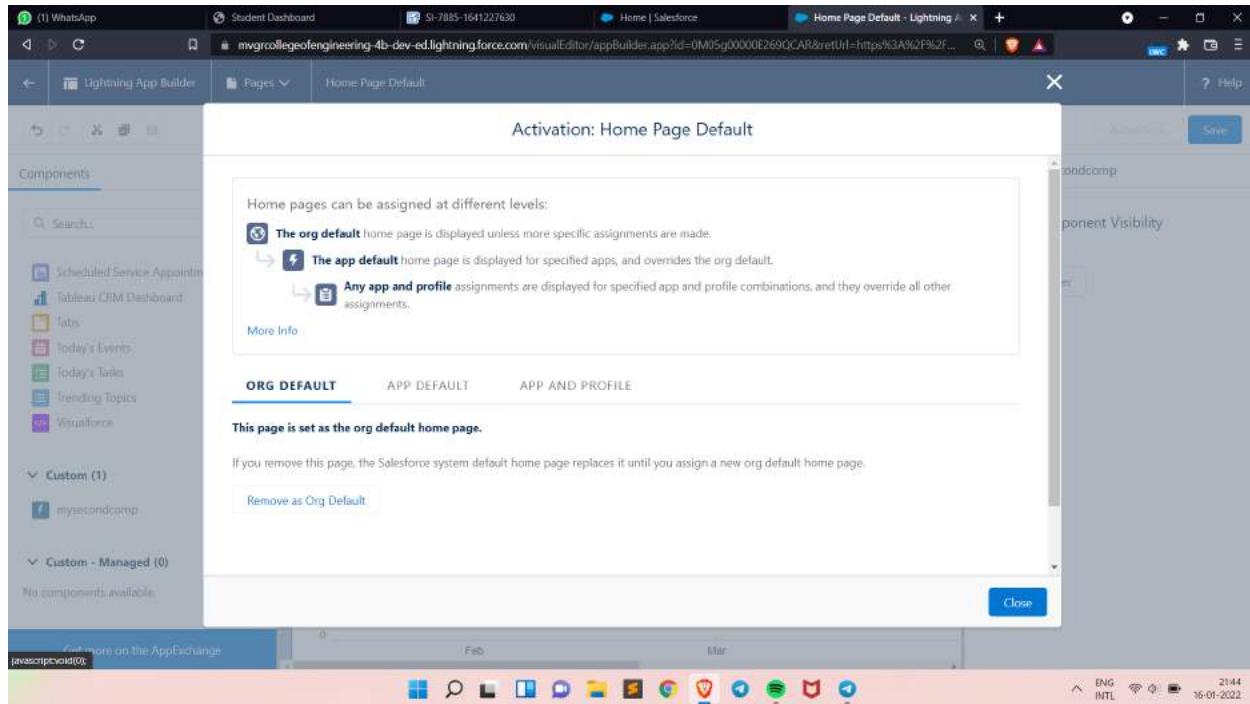
```
21:24:09.334 ended SFDX: Deploy Source to Org
21:26:14.819 Starting SFDX: Deploy Source to Org
...
21:26:18.137 ended SFDX: Deploy Source to Org
21:32:44.867 Starting SFDX: Deploy Source to Org
```
- Output:** Shows deployment status for each file:

STATE	FULL NAME	TYPE	PROJECT PATH
Changed	mysecondcomp	LightningComponentBundle	force-app/main/default/lwc/mysecondcomp/mysecondcomp.css
Changed	mysecondcomp	LightningComponentBundle	force-app/main/default/lwc/mysecondcomp/mysecondcomp.html
Changed	mysecondcomp	LightningComponentBundle	force-app/main/default/lwc/mysecondcomp/mysecondcomp.js
Changed	mysecondcomp	LightningComponentBundle	force-app/main/default/lwc/mysecondcomp/mysecondcomp.js-meta.xml
- Bottom Status Bar:** Shows deployment details: "g:deploy", "In 4, Col 20", "Spaces: 4", "UTF-8", "LF", "XML", "Prettier".

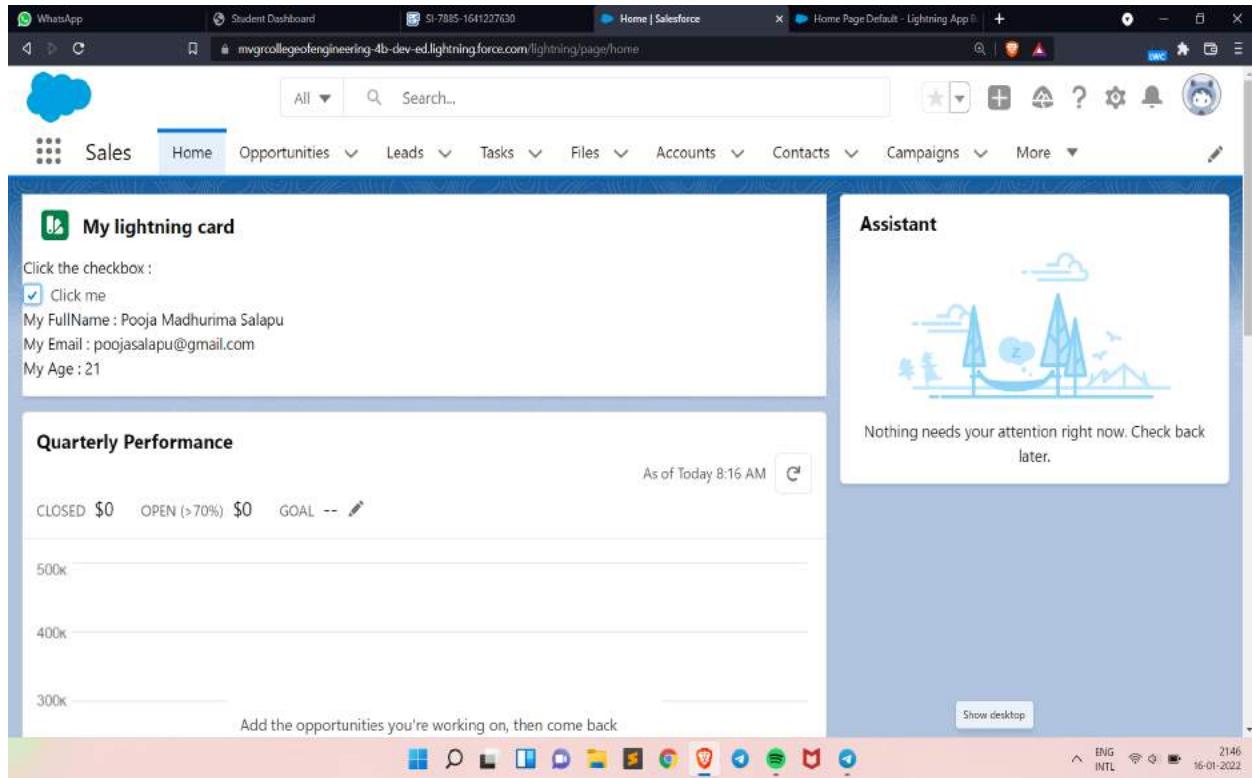
The screenshot shows the Salesforce Lightning App Builder interface with the following details:

- Components Sidebar:** Lists available components: Scheduled Service Appointments, Tableau CRM Dashboard, Tabs, Today's Events, Today's Tasks, Trending Topics, Visualforce, and a custom component "mysecondcomp".
- Page Editor:** Displays a card titled "My lightning card" with the following content:
  - A checkbox labeled "click me".
  - A chart titled "Quarterly Performance" showing revenue trends from Jan to Mar.
  - A message: "Add the opportunities you're working on, then come back here to view your performance."
- Right Panel:** Shows the page path "Page > mysecondcomp" and a "Set Component Visibility" section.
- Bottom Status Bar:** Shows deployment details: "g:deploy", "In 4, Col 20", "Spaces: 4", "UTF-8", "LF", "XML", "Prettier".

# PROJECT REPORT



# PROJECT REPORT



Day 10 :

Topic: Lightning Web Components, iterators and table forms from lwc and apex classes.

Milestone / Activities: To know how to create iterators (for-each loops), child components and parent components and other attributes and deploy it to org. To create a collegeDataTable(html,css,js, and meta file).

Detailed Description: In vs code, create listIterator LWC and we can implement iterator. And another child and parent components which are interconnected. Here the example we did is the string typed being changed to uppercase.

There is another function implemented here which is to search a contact by creating a lwcapex component. Here we used searchcontact functionality and we can see a message called "Button Clicked" whenever we click on search contact button. To see the message, on the web page right click and click on inspect and we can see the message in the javascript console.

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.

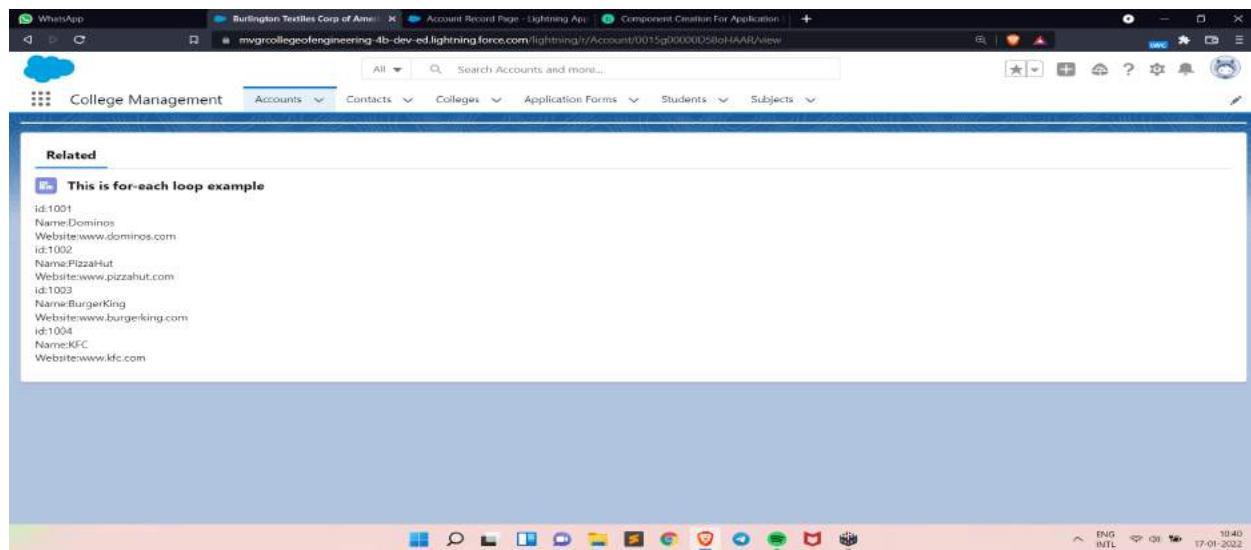
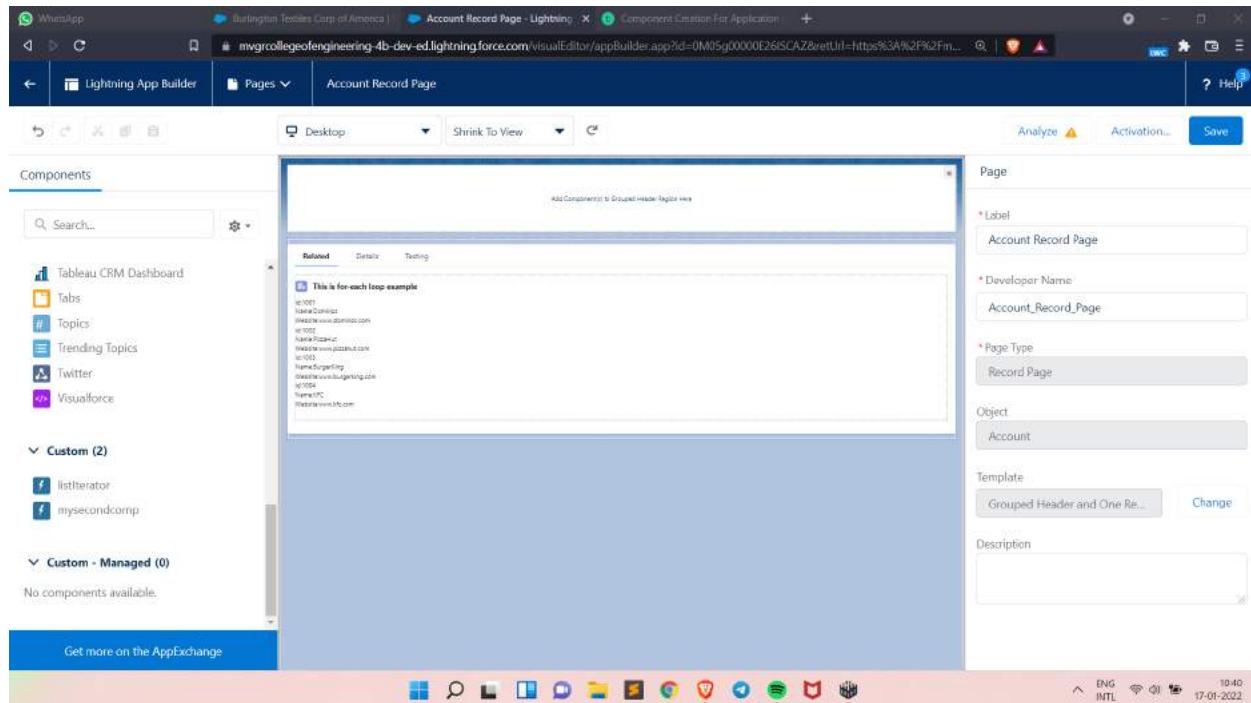
This datatable will reflect in our org when deployed. We also used target as blank and made our form data as a hyperlink. When clicked on the link, a new window opens up

# PROJECT REPORT

and the details of that college will be displayed.

As I didnt create the data before I couldnt get the data reflected in org but the data can be seen in the org.

Upload the Screenshot the Milestone / Activities :



# PROJECT REPORT

Salesforce LWC Editor (Beta)

Search Apex or Component...  listIterator.html  listIterator.js

APEX CLASSES

LIGHTNING WEB COMPONENTS

listIterator

- listIterator.html
- listIterator.js
- listIterator.js-meta.xml

mysecondcamp

listIterator.html

```
<template>
    <lightning-card variant="Narrow" title="This is for-each loop example" icon-name="standard:account">
        <template for:each={contacts} for:item="con" icon-name="standard:contact">
            <div key={con.id}>
                id:{con.id} <br/>
                Name:{con.Name} <br/>
                Website:{con.Website}
            </div>
        </template>
    </lightning-card>
</template>
```

Result Panel

1 listIterator.html successfully deployed source 1/17/2022 10:30:03 AM  
2 listIterator.js successfully deployed source 1/17/2022 10:30:03 AM

MVGR College of Engineering

Use LWC Standard Deployment?  ENG INTL 10:40 17-01-2022

Salesforce LWC Editor (Beta)

Search Apex or Component...  listIterator.html  listIterator.js

APEX CLASSES

LIGHTNING WEB COMPONENTS

listIterator

- listIterator.html
- listIterator.js
- listIterator.js-meta.xml

mysecondcamp

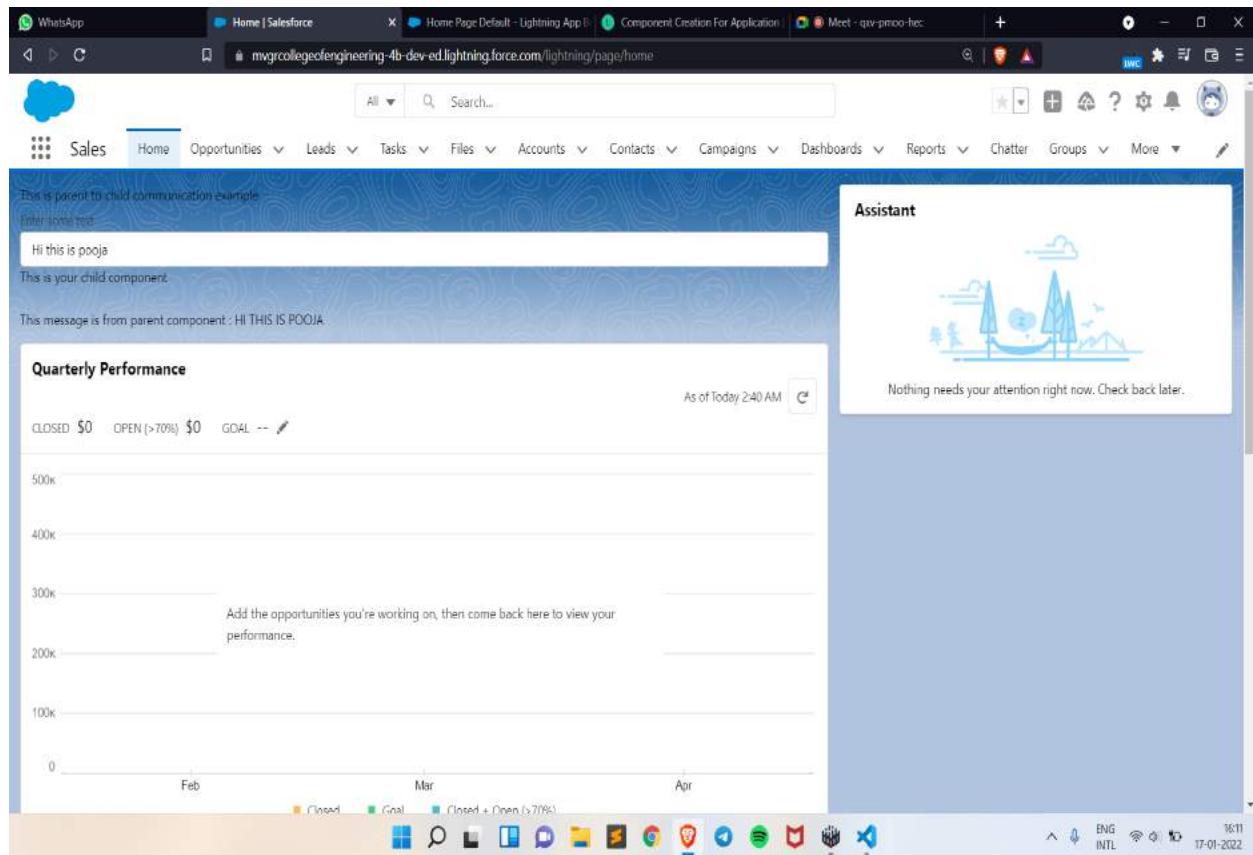
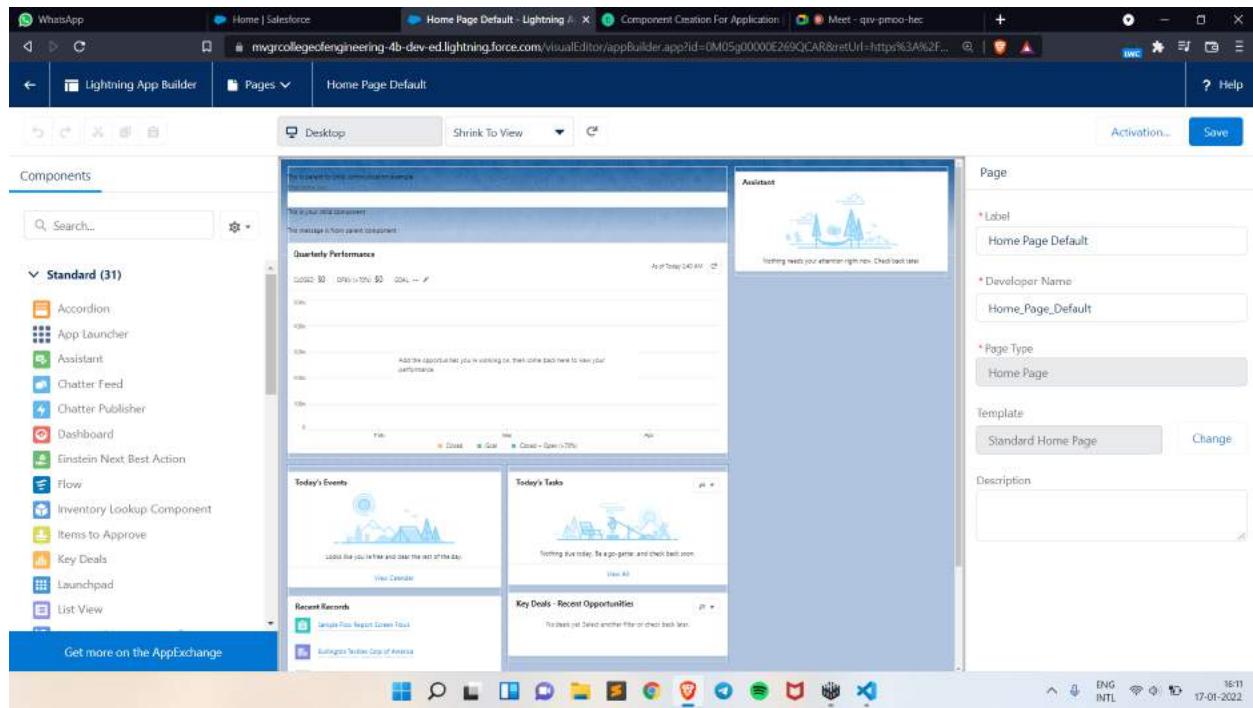
listIterator.js

```
import { LightningElement } from 'lwc';
export default class listIterator extends LightningElement {
    contacts=[
        {
            id:'1001',
            Name:'Dominoes',
            Website:'www.dominos.com'
        },
        {
            id:'1002',
            Name:'PizzaHut',
            Website:'www.pizzahut.com'
        },
        {
            id:'1003',
            Name:'BurgerKing',
            Website:'www.burgerking.com'
        },
        {
            id:'1004',
            Name:'KFC',
            Website:'www.kfc.com'
        }
    ]
}
```

MVGR College of Engineering

Use LWC Standard Deployment?  ENG INTL 10:41 17-01-2022

# PROJECT REPORT



# PROJECT REPORT

File Edit Selection View Go Run Terminal Help

childcmp1.html - collegeManagement - Visual Studio Code

OPEN EDITORS

COLLEGE MANAGEMENT

layouts

lwc

childcmp1

childcmp1.html

childcmp1.js

childcmp1.js-meta.xml

force-app > main > default > lwc > childcmp1 > childcmp1.html > template > h1

```
<template>
<h1>This is your child component</h1><br/>
<h2>This message is from parent component : {message}</h2>
</template>
```

parentcmp1

parentcmp1.html

parentcmp1.js

parentcmp1.js-meta.xml

parentcmp2

jsonconfig.json

objects

permissionsets

staticresources

tabs

triggers

manifest

scripts

apex

hello.apex

sql

account.sql

.eslintignore

OUTLINE

TIMELINE

PLAYGROUND EXAMPLES

RUNNING TASKS

master\*+ 0 0 0 testOrg001 Live Share

globe ENG INTEL 16:38 17-01-2022

File Edit Selection View Go Run Terminal Help

childcmp1.js - collegeManagement - Visual Studio Code

OPEN EDITORS

COLLEGE MANAGEMENT

layouts

lwc

childcmp1

childcmp1.html

childcmp1.js

childcmp1.js-meta.xml

force-app > main > default > lwc > childcmp1 > childcmp1.js > Childcmp1 > message

```
import { LightningElement } from 'lwc';
export default class Childcmp1 extends LightningElement {
    message;
}
```

parentcmp1

parentcmp1.html

parentcmp1.js

parentcmp1.js-meta.xml

parentcmp2

jsonconfig.json

objects

permissionsets

staticresources

tabs

triggers

manifest

scripts

apex

hello.apex

sql

account.sql

.eslintignore

OUTLINE

TIMELINE

PLAYGROUND EXAMPLES

RUNNING TASKS

master\*+ 0 0 0 testOrg001 Live Share

globe ENG INTEL 16:38 17-01-2022

# PROJECT REPORT

The screenshot shows the Visual Studio Code interface with the title bar "childcmp1.js-meta.xml - collegeManagement - Visual Studio Code". The Explorer sidebar on the left shows a project structure under "COLLEGEMANAGEMENT" with various components like layouts, lwc, childcmp1, parentcmp1, and apex scripts. The main editor area displays the XML code for "childcmp1.js-meta.xml":

```
force-app > main > default > lwc > childcmp1 > childcmp1.js-meta.xml
<!DOCTYPE componentbundle xmlns="http://soap.sforce.com/2006/04/metadata">
<componentVersion>52.0</componentVersion>
<isExposed>false</isExposed>
</LightningComponentBundle>
```

The status bar at the bottom shows "gplocate Ln 1, Col 1 Spaces: 4 UTF-8 LF XML ⚡ Prettier" and the system tray indicates "16:38 INTEL 17-01-2022".

The screenshot shows the Visual Studio Code interface with the title bar "parentcmp1.html - collegeManagement - Visual Studio Code". The Explorer sidebar on the left shows the same project structure. The main editor area displays the HTML code for "parentcmp1.html":

```
<template>
<!--this is parent to child communication example-->
<lightning-input type="text" label="Enter some text" onchange=[handleChange]></lightning-input>
<c-childcmp1 message=(parentmsg)></c-childcmp1>
<img alt=""/>
```

The status bar at the bottom shows "gplocate Ln 5, Col 12 Spaces: 4 UTF-8 LF HTML ⚡ Prettier" and the system tray indicates "16:38 INTEL 17-01-2022".

# PROJECT REPORT

The screenshot shows the Visual Studio Code interface with the following details:

- File Path:** force-app/main/default/lwc/parentcmp1
- File Opened:** Parentcmp1.js
- Code Content:**

```
1 import { LightningElement } from 'lwc';
2
3 export default class Parentcmp1 extends LightningElement {
4     parentmsg;
5     handlechange(event){
6         this.parentmsg.event.target.value;
7         this.parentmsg=this.parentmsg.toUpperCase();
8     }
9 }
```

- Explorer View:** Shows the project structure under COLLEGEMANAGEMENT, including layouts, lwc components (childcmp1, childcmp2), mysecondcomp, parentcmp1, parentcmp2, jsonconfig.json, objects, permissionsets, staticresources, tabs, triggers, manifest, scripts (hello.apex, account.sql), and .eslintignore.
- Bottom Status Bar:** gitlocate Ln 9, Col 2 Spaces: 4 UTF-8 LF JavaScript Prettier
- System Tray:** ENG INTEL 16:39 17-01-2022

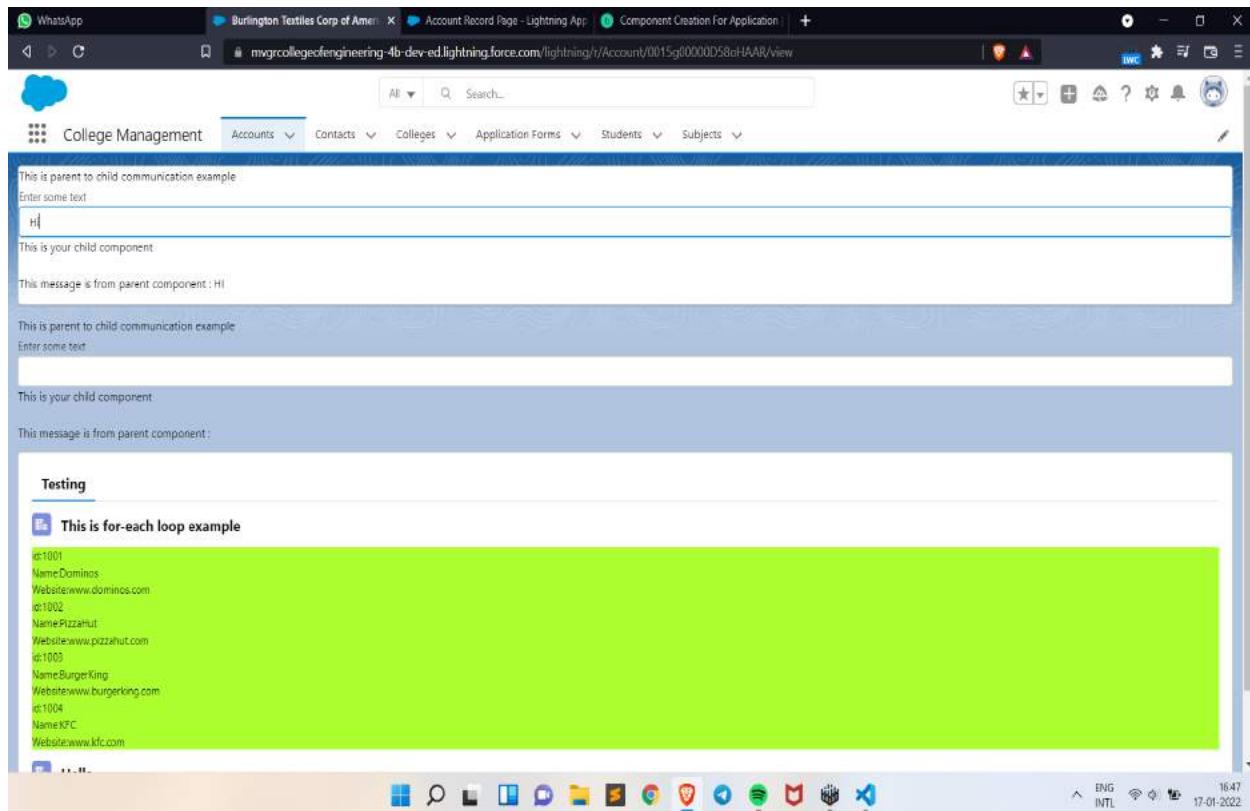
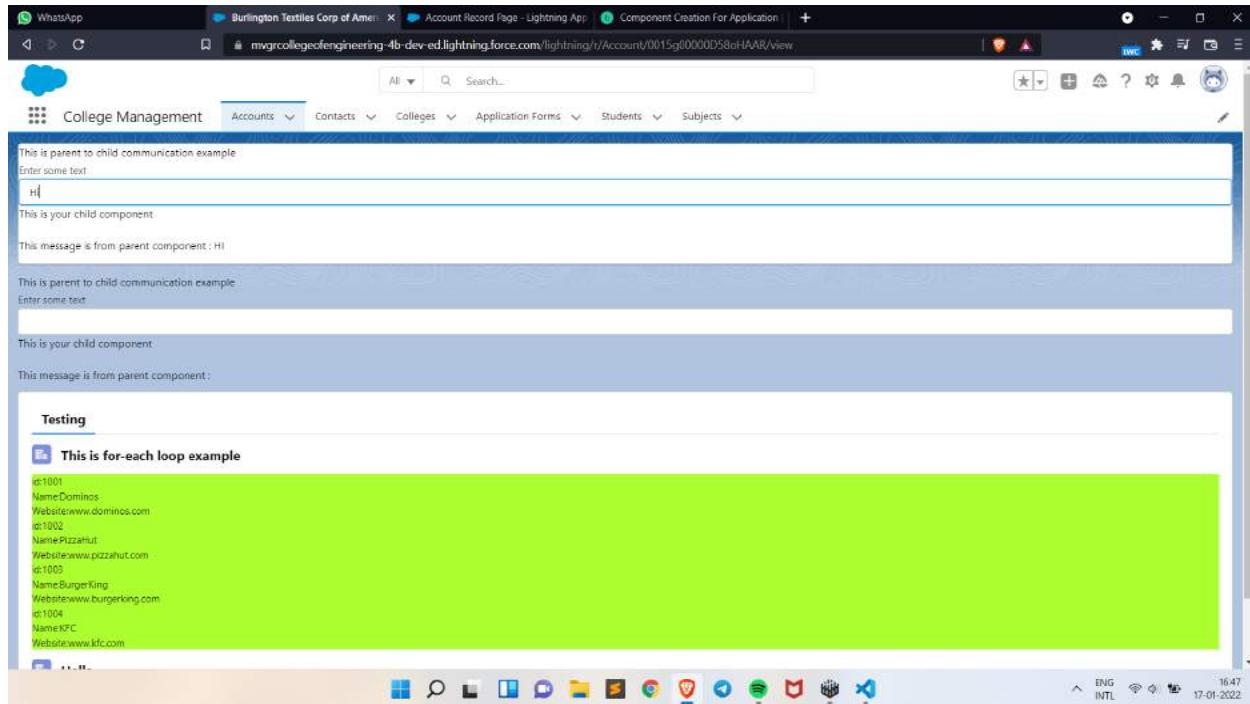
The screenshot shows the Visual Studio Code interface with the following details:

- File Path:** force-app/main/default/lwc/parentcmp1
- File Opened:** parentcmp1.js-meta.xml
- Code Content:**

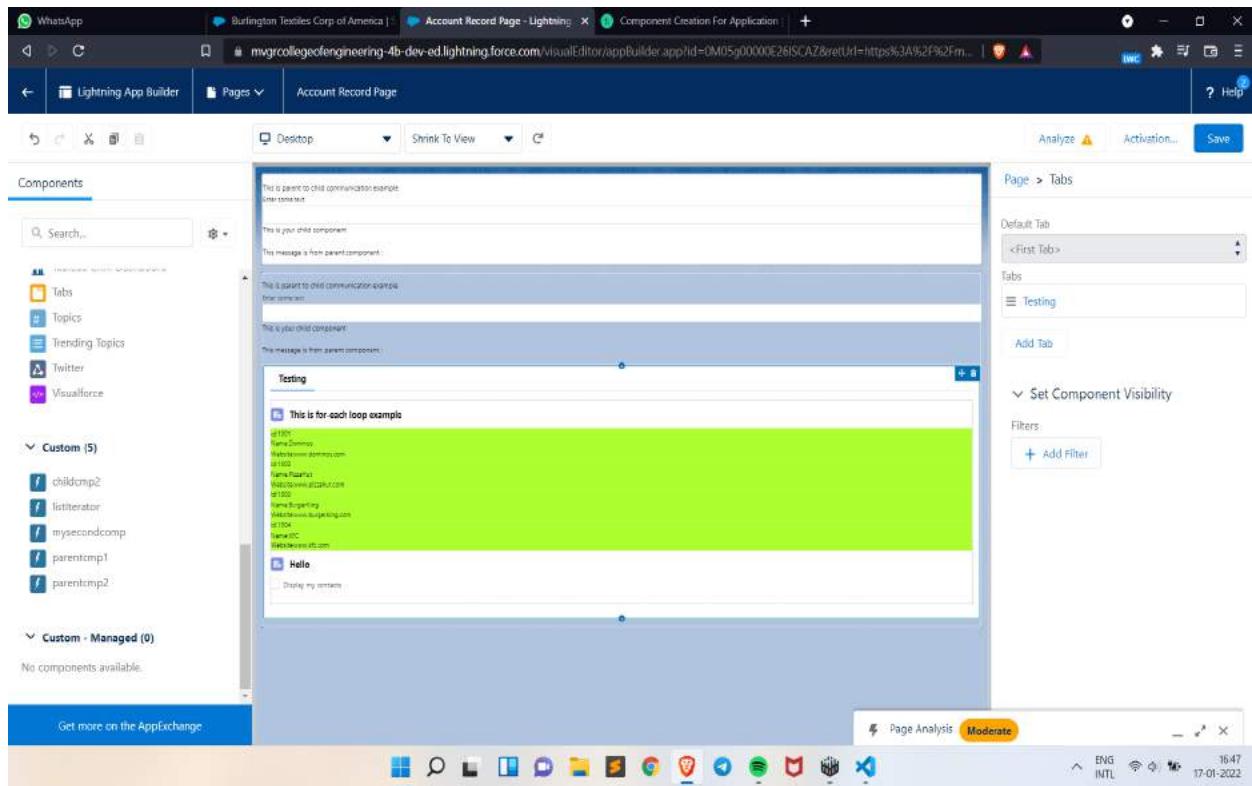
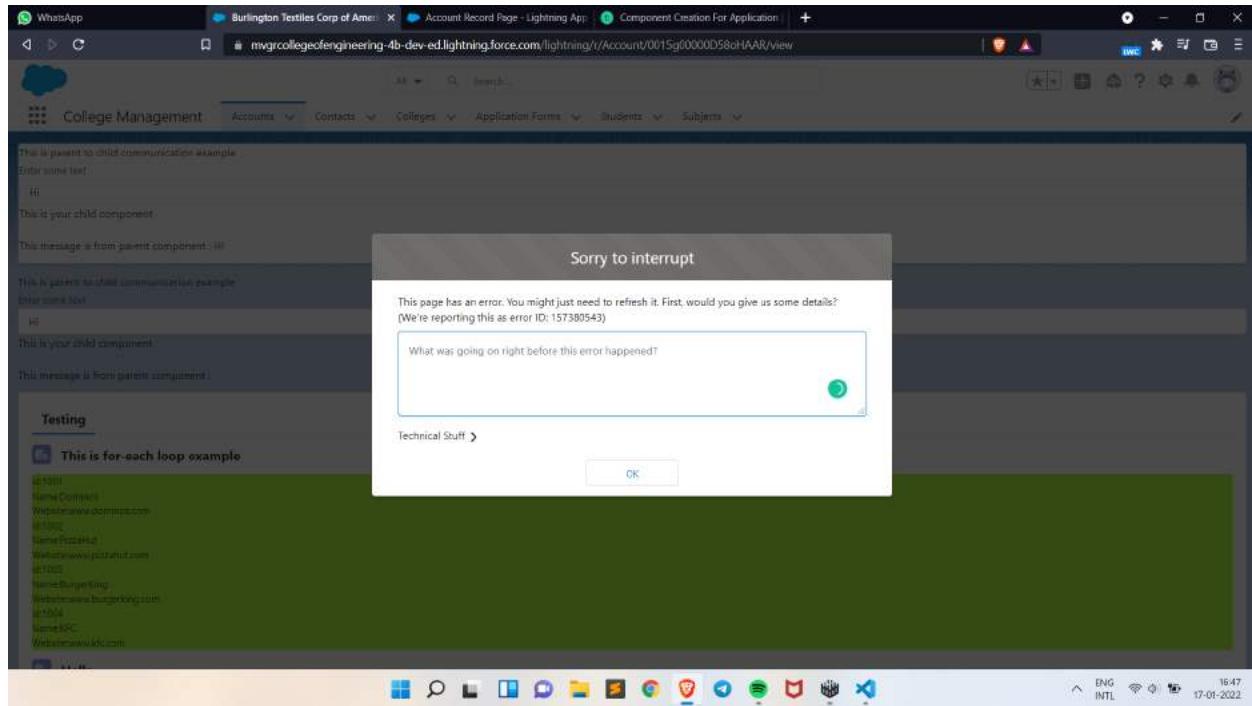
```
1 <xml version="1.0" encoding="UTF-8">
2 <LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
3     <apiVersion>52.0</apiVersion>
4     <isExposed>true</isExposed>
5     <targets>
6         <target>lightning_RecordPage</target>
7         <target>lightning_AppPage</target>
8         <target>lightning_HomePages</target>
9     </targets>
10 </LightningComponentBundle>
```

- Explorer View:** Shows the project structure under COLLEGEMANAGEMENT, including layouts, lwc components (childcmp1, childcmp2), mysecondcomp, parentcmp1, parentcmp2, jsonconfig.json, objects, permissionsets, staticresources, tabs, triggers, manifest, scripts (hello.apex, account.sql), and .eslintignore.
- Bottom Status Bar:** gitlocate Ln 9, Col 15 Spaces: 4 UTF-8 LF XML Prettier
- System Tray:** ENG INTEL 16:39 17-01-2022

# PROJECT REPORT



# PROJECT REPORT



# PROJECT REPORT

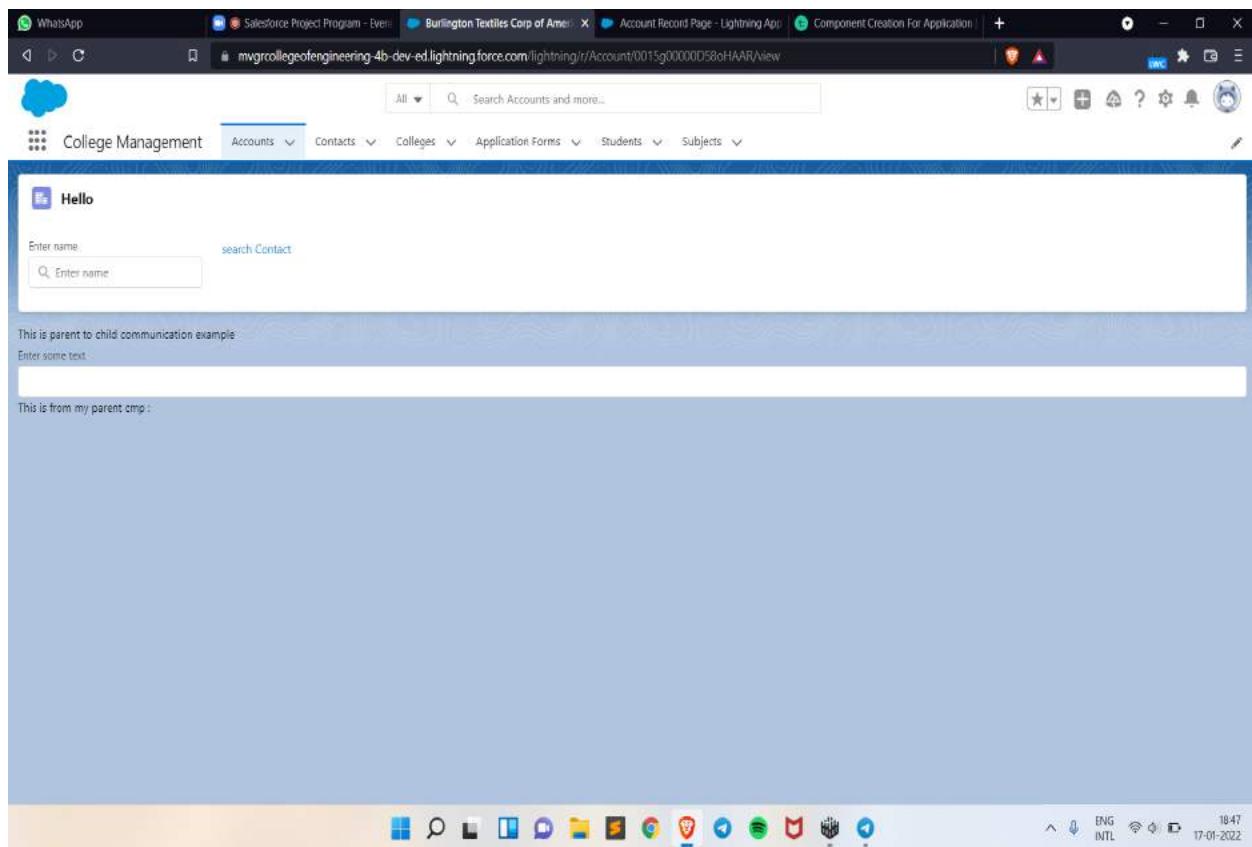
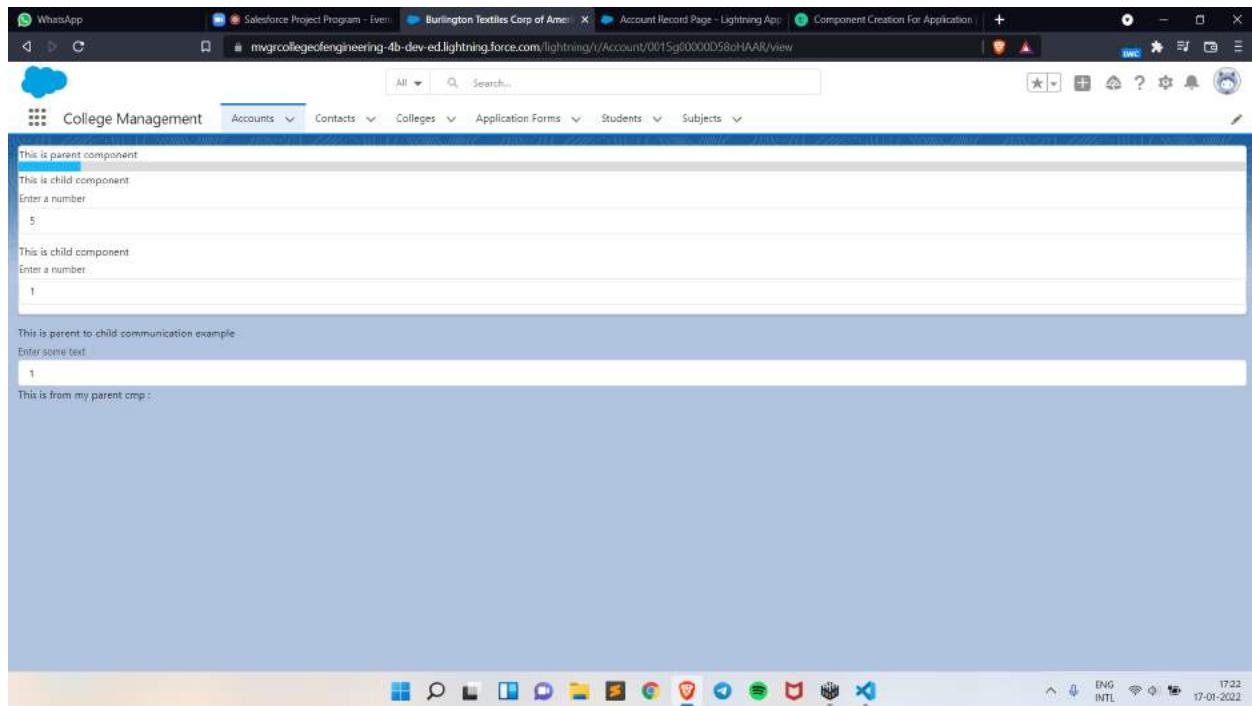
The screenshot shows the Visual Studio Code interface with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Explorer:** Shows a tree view of files under "COLLEGEMANAGEMENT".
- Code Editor:** Displays a component definition file (parentcmp2.html) with the following code:<template>
<div>This is parent to child communication example</div>
<lightning-input type="text" label="Enter some text" onchange={handleChange}></lightning-input>
<childcmp2></childcmp2>
- Terminal:** Shows the output of a deployment command: "16:48:43.382 ended SFOX: Deploy Source to Org".
- Bottom Status Bar:** Shows file location, line 5, column 12, spaces 4, UTF-8, LF, HTML, and Prettier icons.

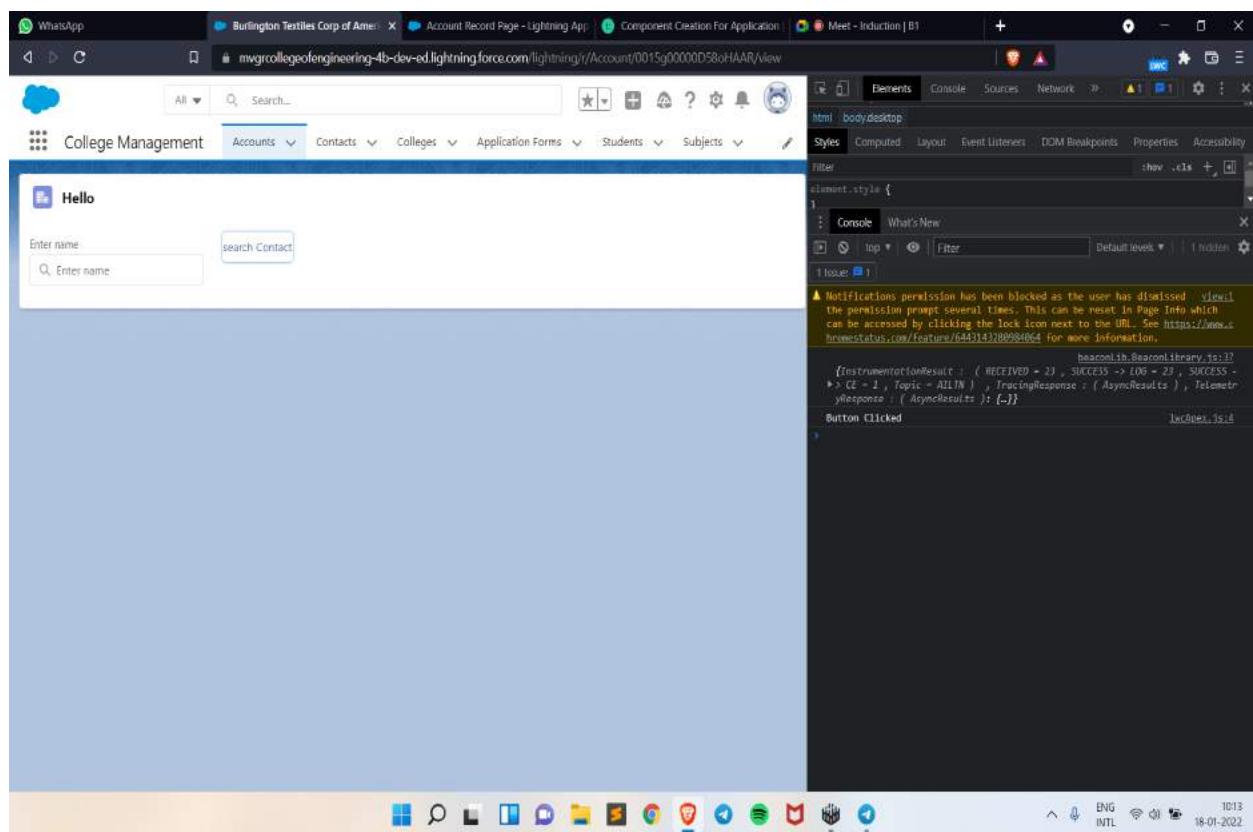
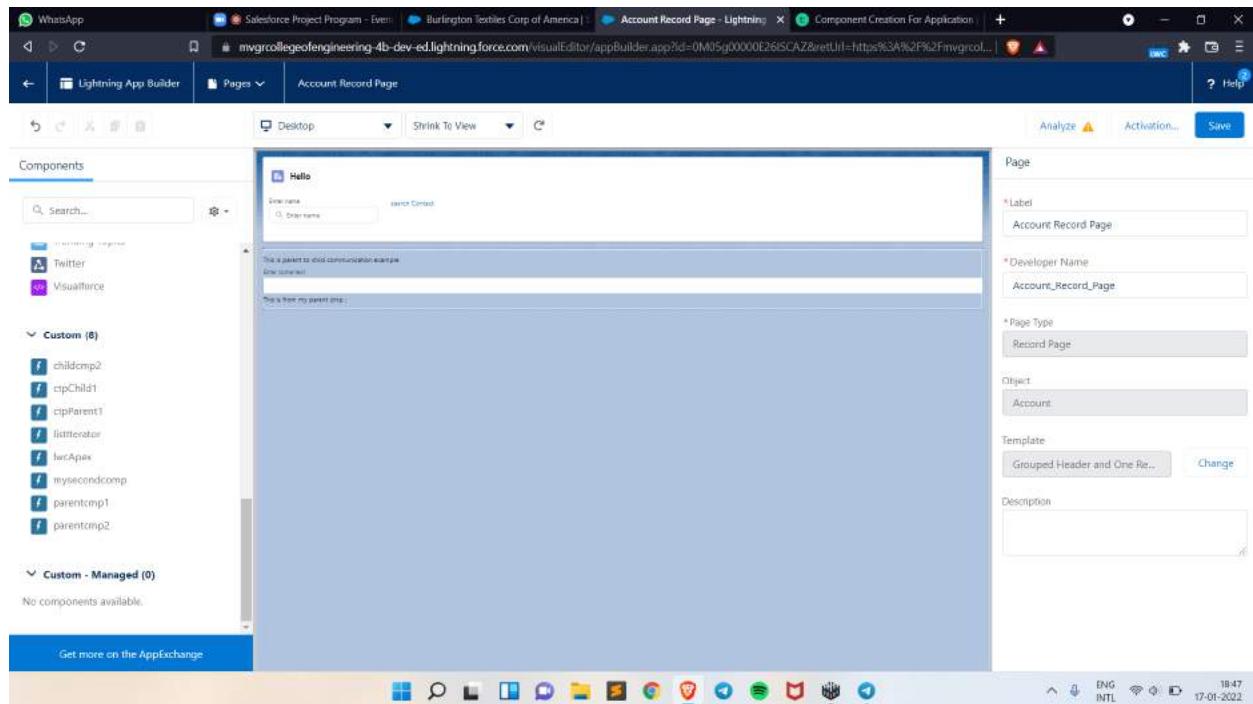
The screenshot shows the Salesforce Lightning App Builder interface with the following details:

- Header:** WhatsApp, Salesforce Project Program - Event, Burlington Textiles Corp of America, Account Record Page - Lightning, Component Creation For Application.
- Left Sidebar:** Components, Search bar, and links for LinkedIn, Trending Topics, Twitter, and Visualforce.
- Central Area:** A component creation dialog for "parentcmp2" with sections for "File to parent component", "This is child component", "This is parent to child communication example", and "This is from my parent component".
- Right Sidebar:** Set Component Visibility, Filters, and Add Filter button.
- Bottom Navigation:** Page Analysis (Moderate), Get more on the AppExchange, and system status icons.

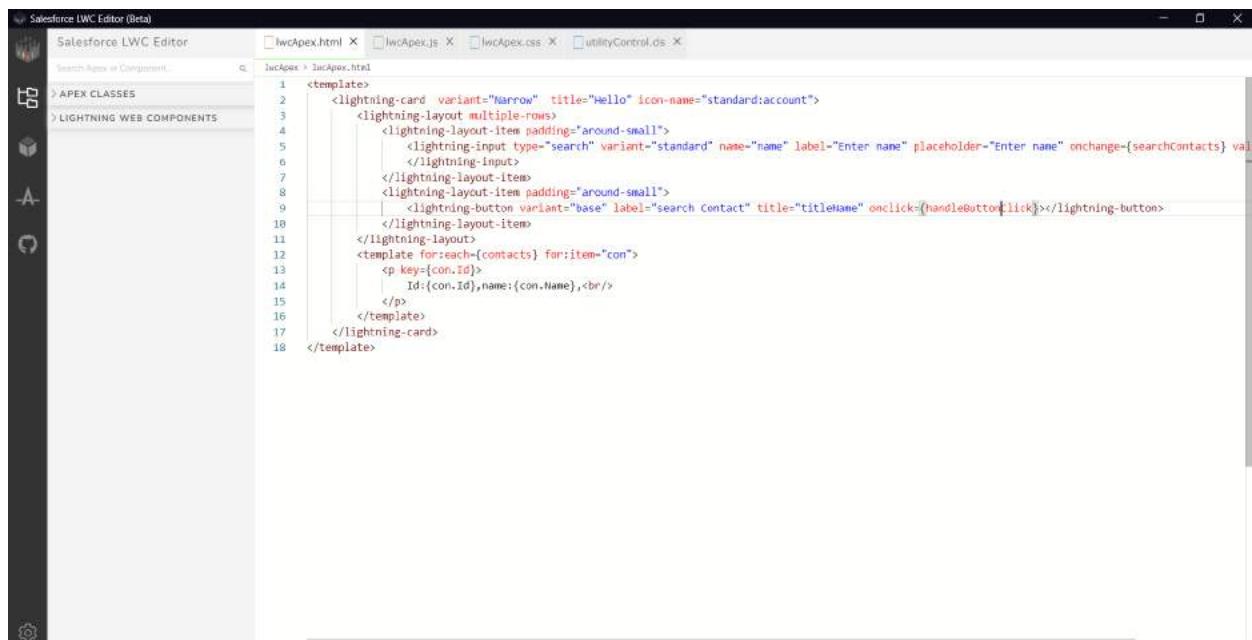
# PROJECT REPORT



# PROJECT REPORT

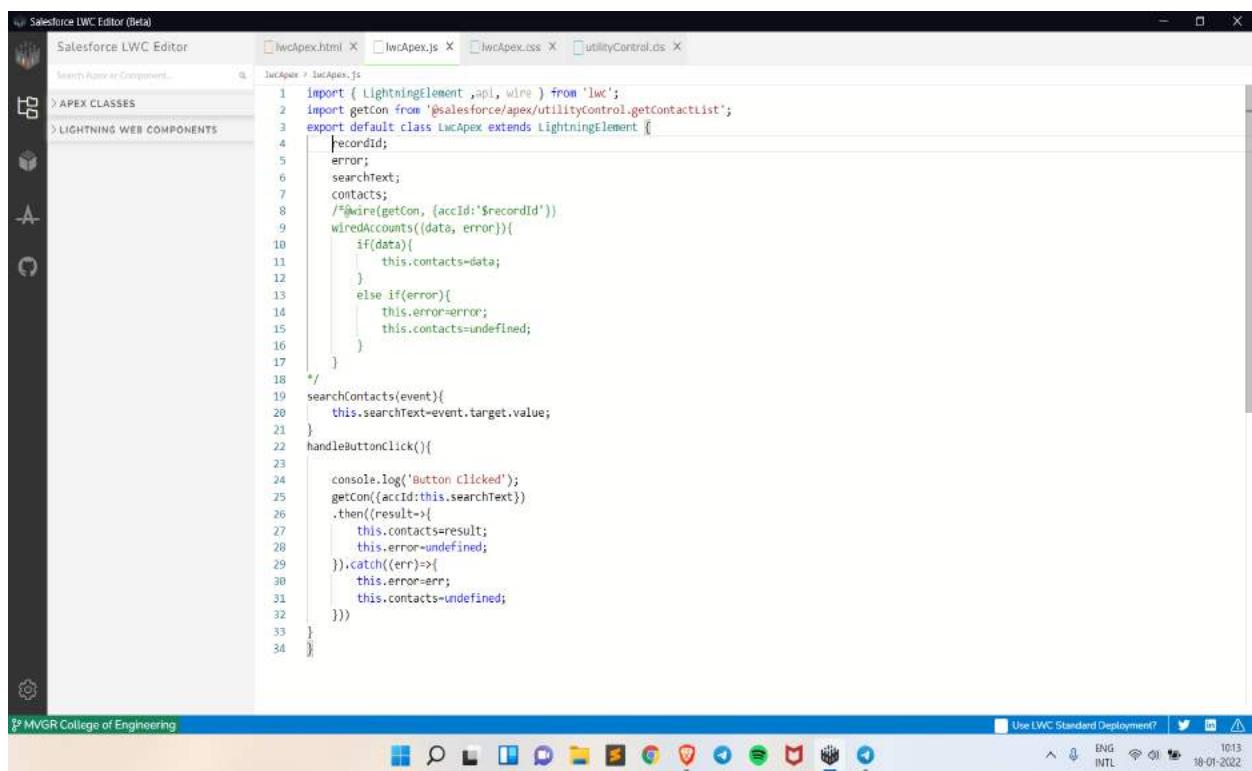


# PROJECT REPORT



The screenshot shows the Salesforce LWC Editor interface. The top navigation bar includes tabs for 'Salesforce LWC Editor (Beta)', 'lwcApex.html', 'lwcApex.js', 'lwcApex.css', and 'utilityControl.ds'. On the left, a sidebar lists 'Search Apex or Component...', 'APEX CLASSES', and 'LIGHTNING WEB COMPONENTS'. The main content area displays the code for the template file 'lwcApex.html'.

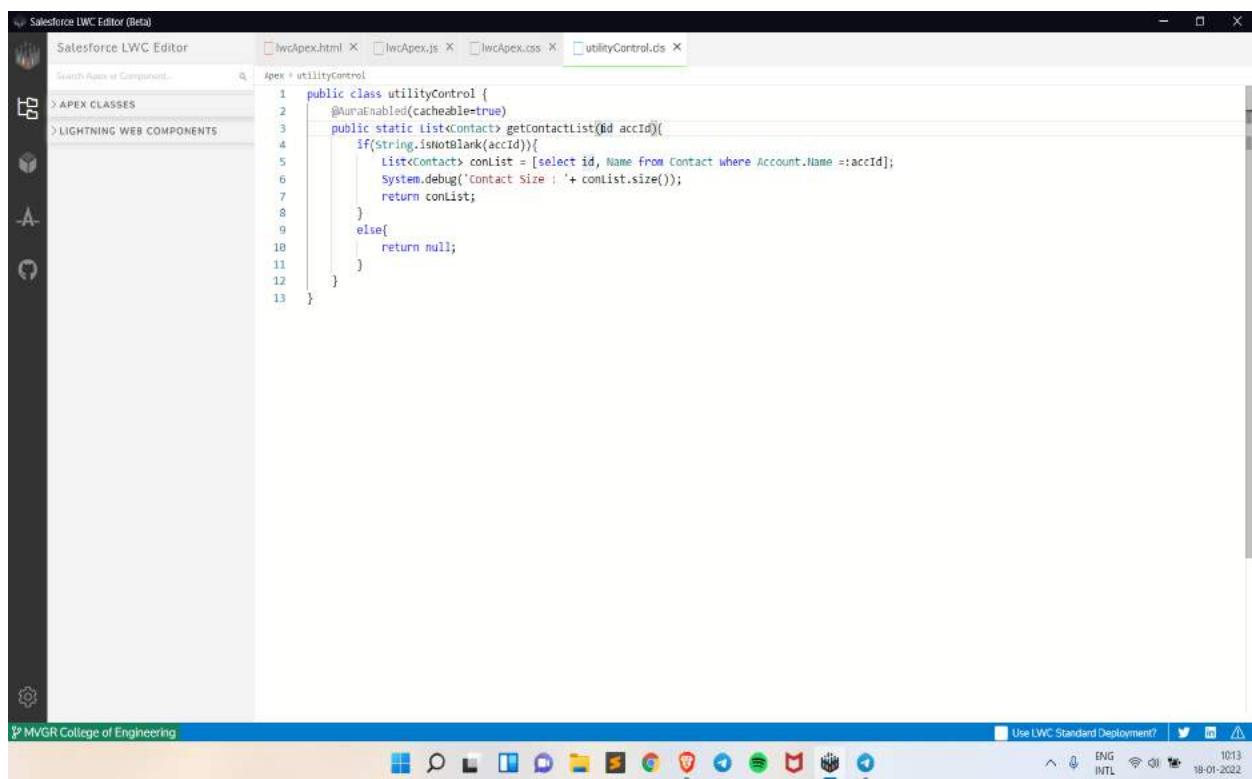
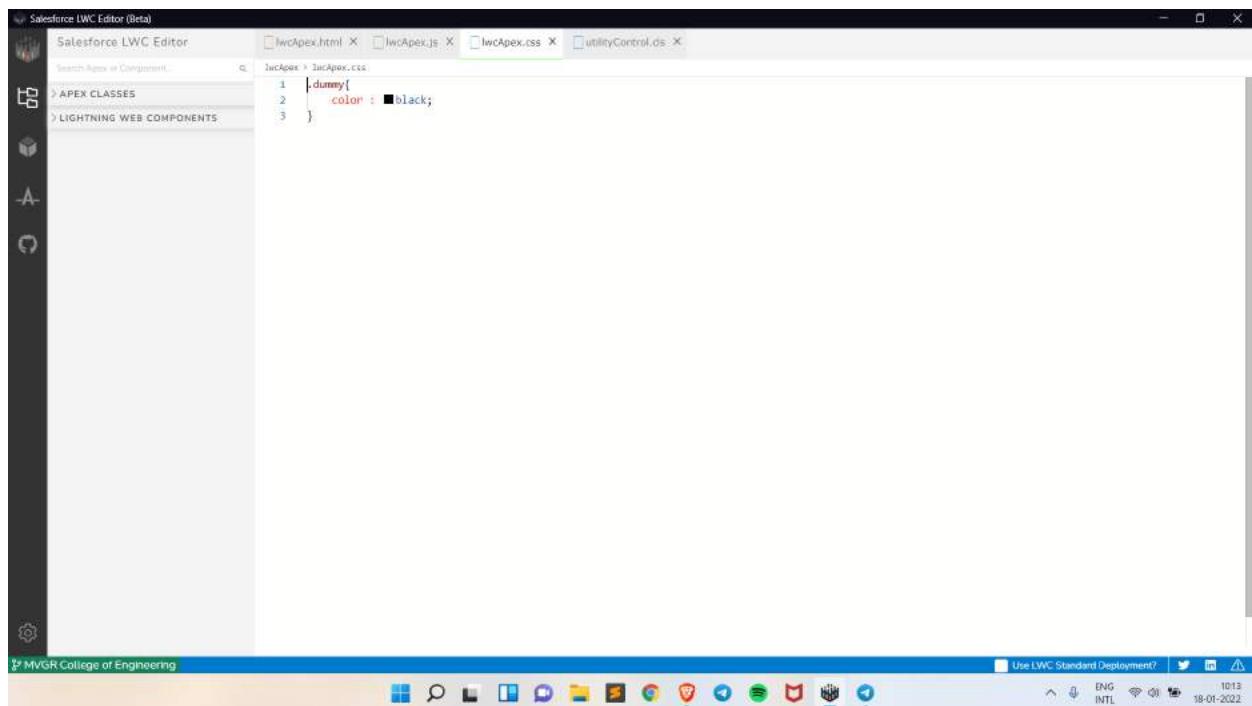
```
<template>
    <lightning-card variant="Narrow" title="Hello" icon-name="standard:account">
        <lightning-layout multiple-rows>
            <lightning-layout-item padding="around-small">
                <lightning-input type="search" variant="standard" name="name" label="Enter name" placeholder="Enter name" onchange={searchContacts} value=""/>
            </lightning-layout-item>
            <lightning-layout-item padding="around-small">
                <lightning-button variant="base" label="search Contact" title="titleName" onclick={handleButtonclick}></lightning-button>
            </lightning-layout-item>
        </lightning-layout>
        <template for:each={contacts} for:item="con">
            <p key={con.Id}>
                Id:{con.Id}, Name:{con.Name}, <br/>
            </p>
        </template>
    </lightning-card>
</template>
```



The screenshot shows the Salesforce LWC Editor interface. The top navigation bar includes tabs for 'Salesforce LWC Editor (Beta)', 'lwcApex.html', 'lwcApex.js', 'lwcApex.css', and 'utilityControl.ds'. On the left, a sidebar lists 'Search Apex or Component...', 'APEX CLASSES', and 'LIGHTNING WEB COMPONENTS'. The main content area displays the code for the JavaScript file 'lwcApex.js'.

```
import { LightningElement, api, wire } from 'lwc';
import getCon from '@salesforce/apex/utilityControl/getContactList';
export default class LwcApex extends LightningElement {
    recordId;
    error;
    searchText;
    contacts;
    /*@wire(getCon, {accId:$recordId})
    wiredAccounts({data, error}){*/
        if(data){
            this.contacts=data;
        }
        else if(error){
            this.error=error;
            this.contacts=undefined;
        }
    }
    searchContacts(event){
        this.searchText=event.target.value;
    }
    handleButtonClick(){
        console.log('Button Clicked');
        getCon({accId:this.searchText})
        .then(result=>{
            this.contacts=result;
            this.error=undefined;
        }).catch((err)=>{
            this.error=err;
            this.contacts=undefined;
        })
    }
}
```

# PROJECT REPORT



# PROJECT REPORT

Salesforce LWC Editor (Beta)

Search Apex or Component... APEX CLASSES LIGHTNING WEB COMPONENTS collegeDataTable childcmp1 childcmp2 cptChild1 cptParent1 listIterator lwcApex lwcApex.html lwcApex.js lwcApex.css lwcApex.js-meta.xml mysecondcomp parentcmp1 parentcmp2

```
utilityControl.cls X collegeDataTable.html X collegeDataTable.js X collegeDataTable.css X collegeDataTable.js-meta.xml X
APEX > utilityControl
1  public class utilityControl {
2    @AuraEnabled(cacheable=true)
3    public static List<Contact> getContactList(String accId){
4      if(String.isNotBlank(accId)){
5        List<Contact> contactList = [select id, Name from Contact where Account.Name =:accId];
6        System.debug('contact size : '+ contactList.size());
7        return contactList;
8      }
9    else{
10
11  }
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
}
public class formWrapper{
  @AuraEnabled public String Name {get;set;}
  @AuraEnabled public String student_Name {get;set;}
  @AuraEnabled public String Guardian_Name {get;set;}
  @AuraEnabled public Decimal College_fees {get;set;}}

utilityControl.cls X collegeDataTable.html X collegeDataTable.js X collegeDataTable.css X collegeDataTable.js-meta.xml X
Use LWC Standard Deployment? 12:17 18-01-2022
```

MVGR College of Engineering

Salesforce LWC Editor (Beta)

Search Apex or Component... APEX CLASSES LIGHTNING WEB COMPONENTS collegeDataTable childcmp1 childcmp2 cptChild1 cptParent1 listIterator lwcApex lwcApex.html lwcApex.js lwcApex.css lwcApex.js-meta.xml mysecondcomp parentcmp1 parentcmp2

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

utilityControl.cls X collegeDataTable.html X collegeDataTable.js X collegeDataTable.css X collegeDataTable.js-meta.xml X
Use LWC Standard Deployment? 12:17 18-01-2022
```

MVGR College of Engineering

# PROJECT REPORT

Salesforce LWC Editor (Beta)

Search Apex or Component...

> APEX CLASSES

> LIGHTNING WEB COMPONENTS

collegeDataTable

- collegeDataTable.html
- collegeDataTable.js
- collegeDataTable.css
- collegeDataTable.js-meta.xml

childcmp1

childcmp2

ctpChild1

ctpParent1

lwcApex

- lwcApex.html
- lwcApex.js
- lwcApex.css
- lwcApex.js-meta.xml

mysecondcomp

parentcmp1

parentcmp2

UtilityControl.cls X collegeDataTable.html X collegeDataTable.js X collegeDataTable.css X collegeDataTable.js-meta.xml X

```
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 }
```

MVGR College of Engineering

Use LWC Standard Deployment?  ENG INTL 12:17 18-01-2022

Salesforce LWC Editor (Beta)

Search Apex or Component...

> APEX CLASSES

> LIGHTNING WEB COMPONENTS

collegeDataTable

- collegeDataTable.html
- collegeDataTable.js
- collegeDataTable.css
- collegeDataTable.js-meta.xml

childcmp1

childcmp2

ctpChild1

ctpParent1

listIterator

lwcApex

- lwcApex.html
- lwcApex.js
- lwcApex.css
- lwcApex.js-meta.xml

mysecondcomp

parentcmp1

parentcmp2

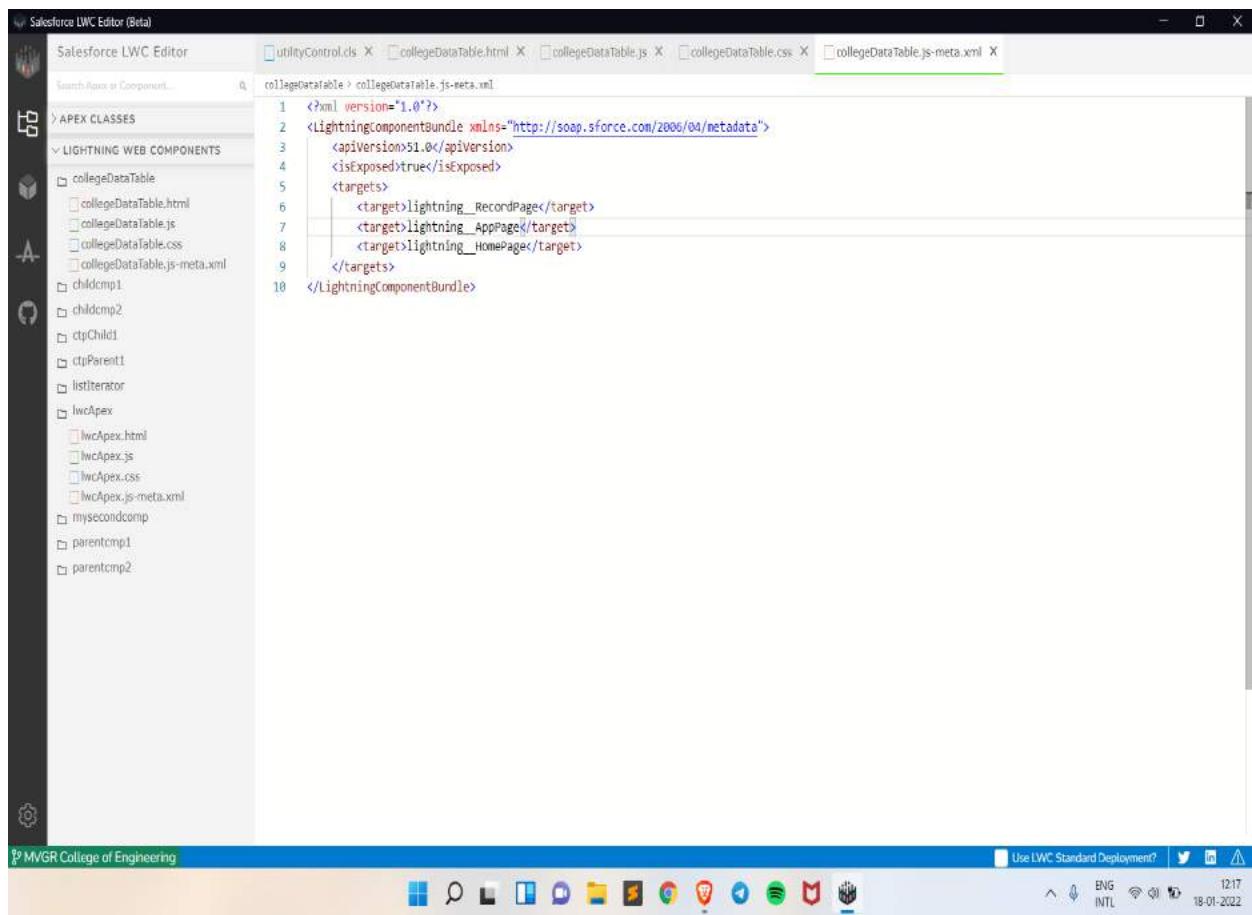
UtilityControl.cls X collegeDataTable.html X collegeDataTable.js X collegeDataTable.css X collegeDataTable.js-meta.xml X

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

MVGR College of Engineering

Use LWC Standard Deployment?  ENG INTL 12:17 18-01-2022

# PROJECT REPORT



Day 11 :

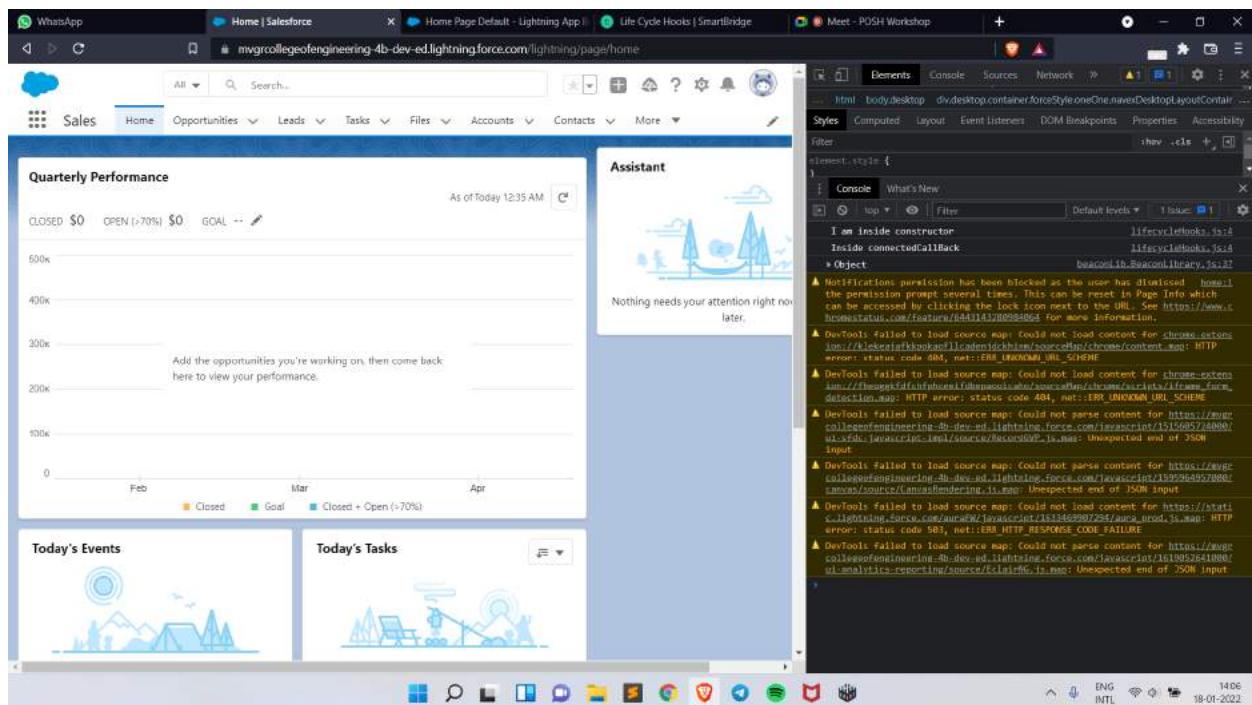
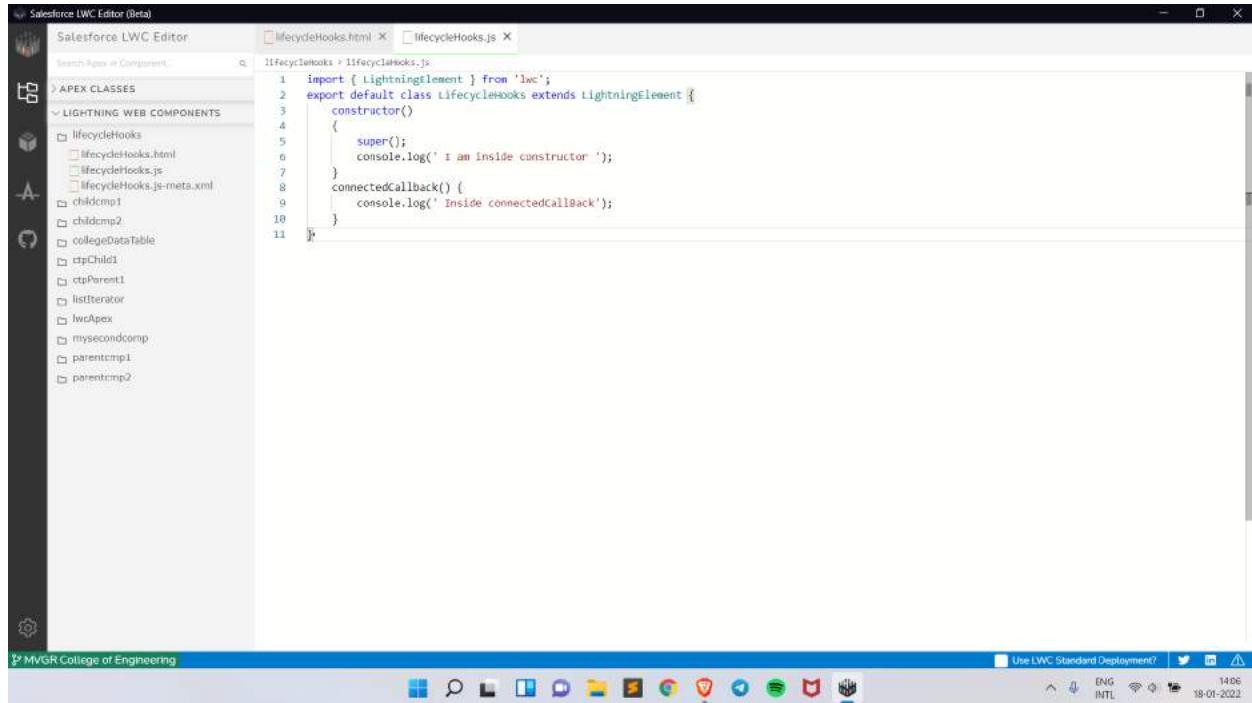
Topic: Alerts in LWC, Security Settings.

Milestone / Activities: To create alerts in LWC.

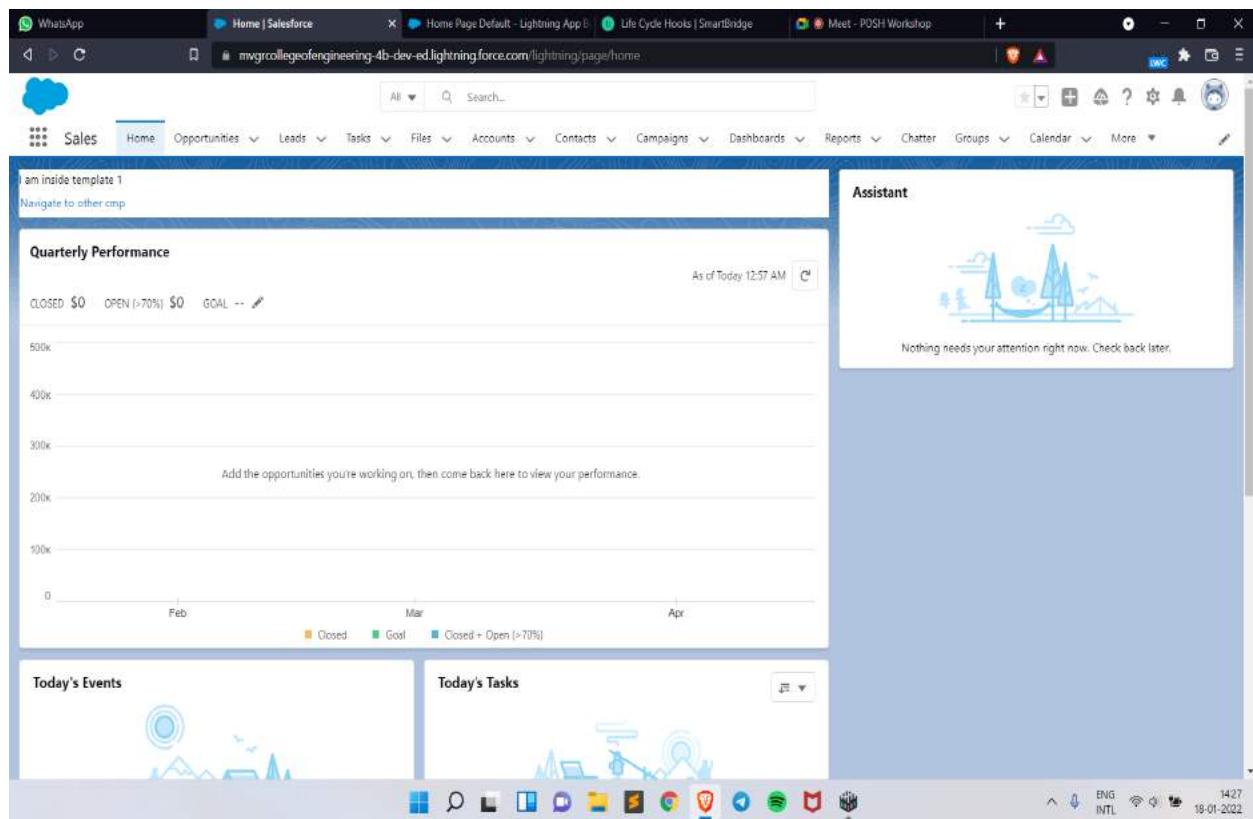
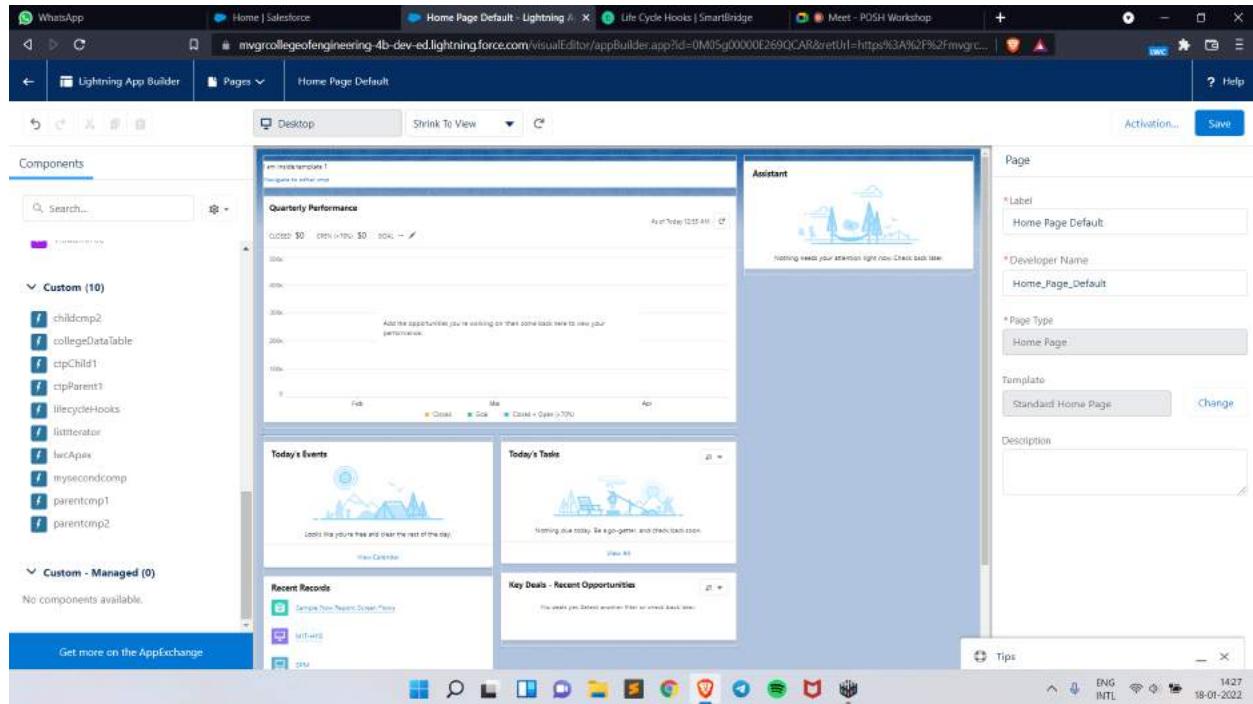
Detailed Description: Create a lwc. Example if we want to create an alert message for a phone number, we can create functions with certain conditions. Success, Error, Warning and Info functions can be implemented.

# PROJECT REPORT

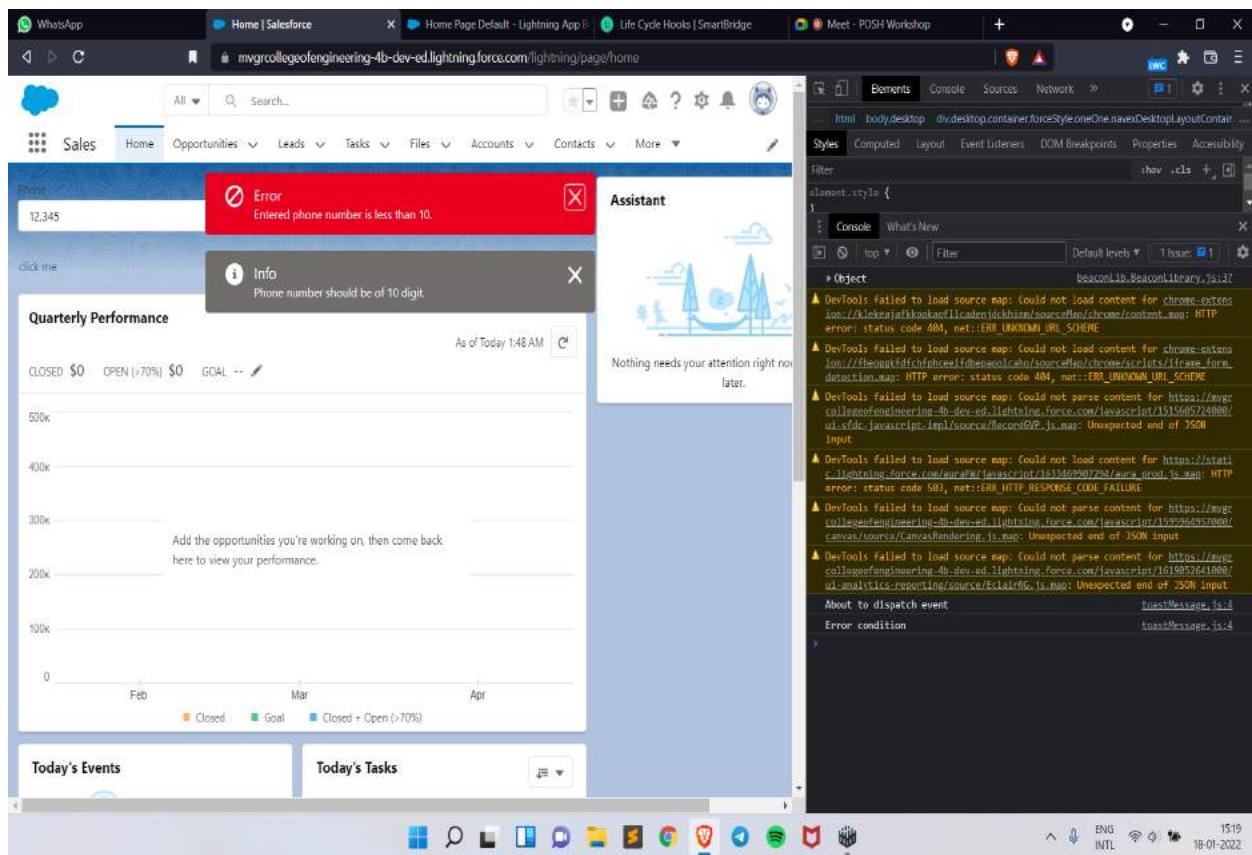
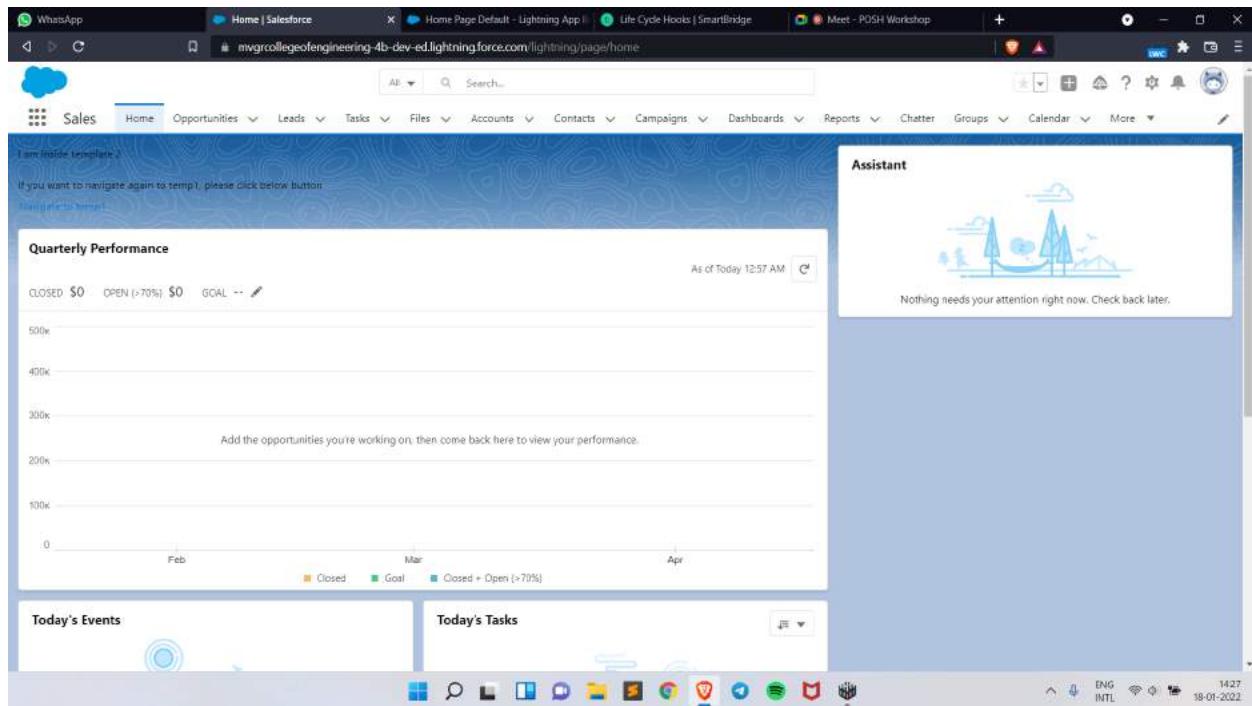
Upload the Screenshot the Milestone / Activities :



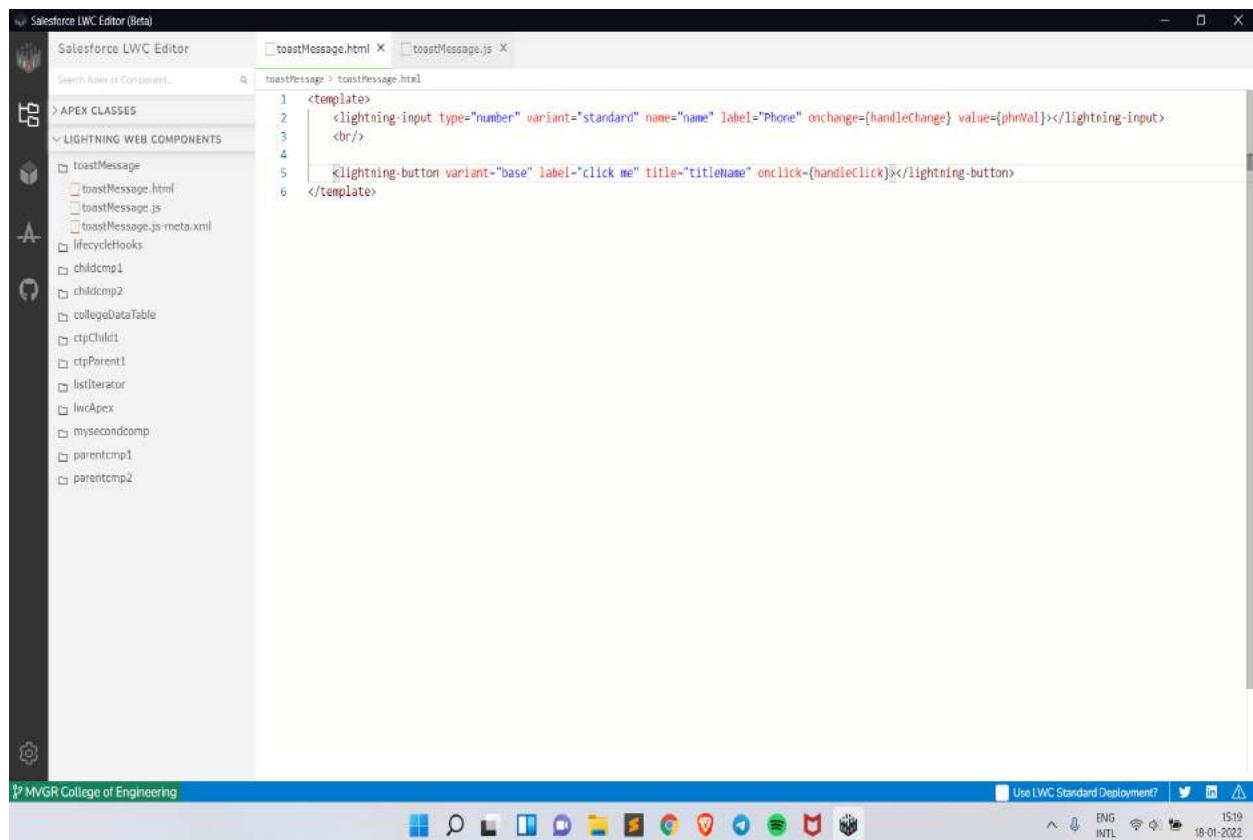
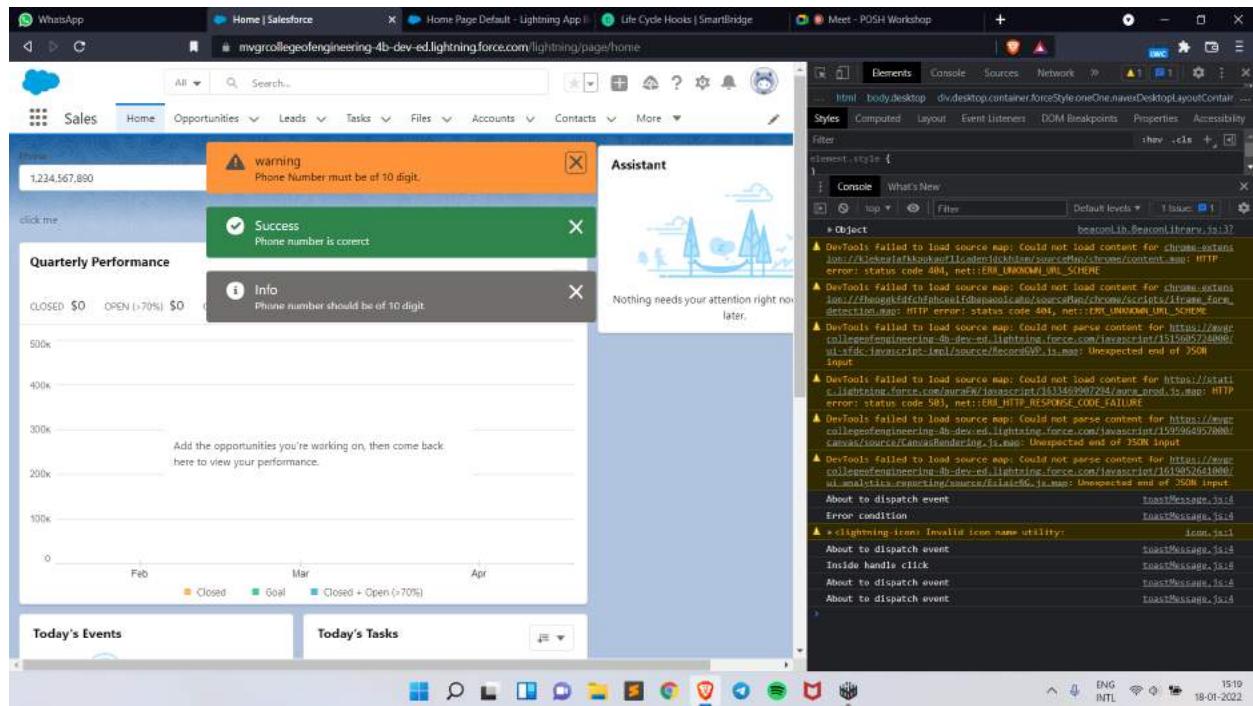
# PROJECT REPORT



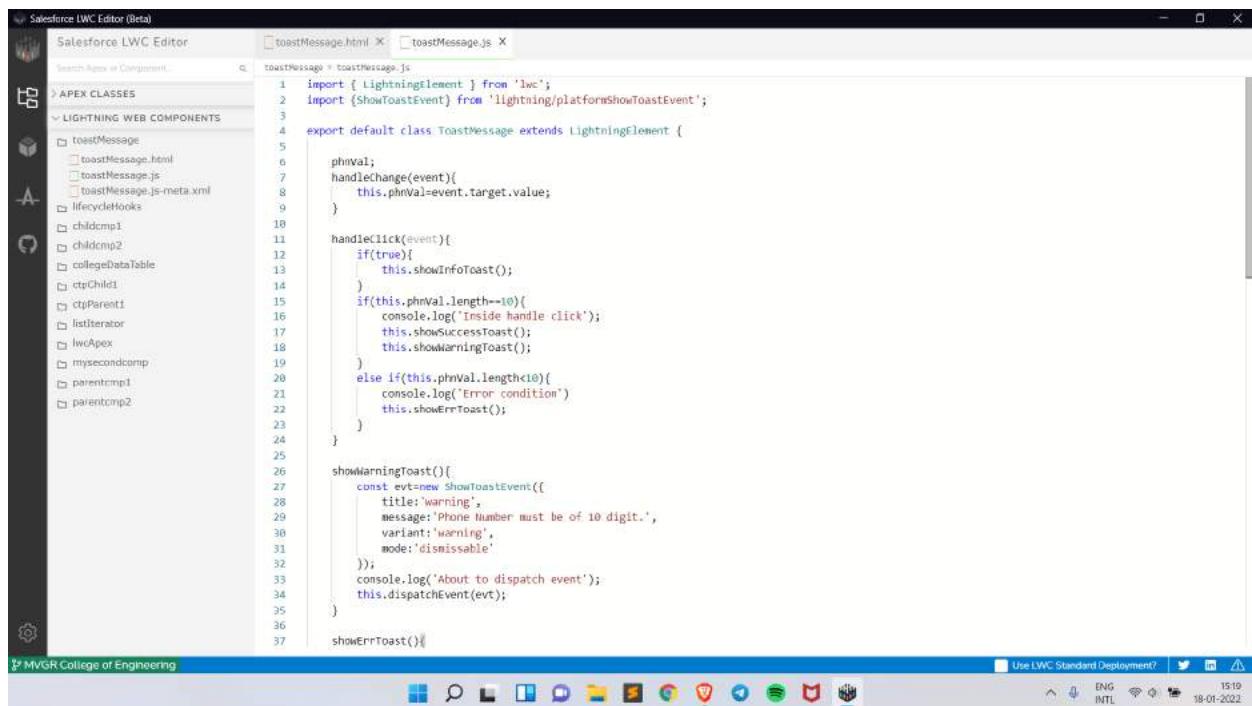
# PROJECT REPORT



# PROJECT REPORT

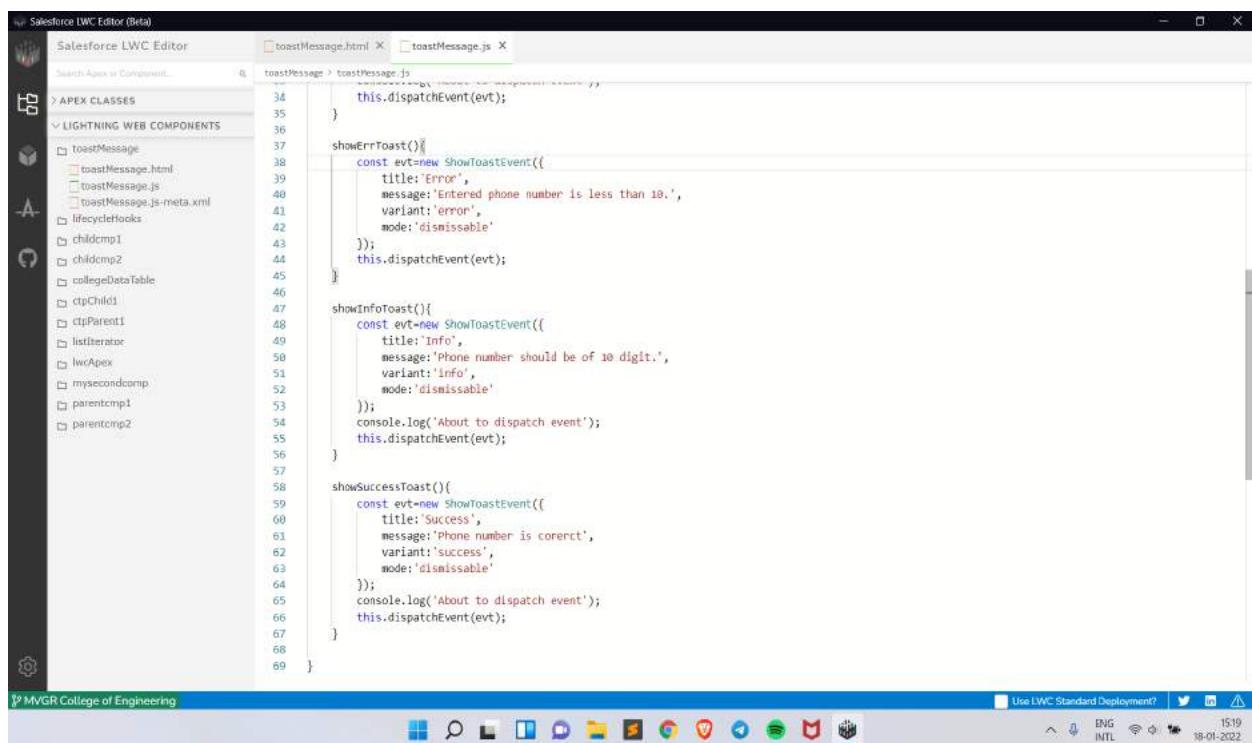


# PROJECT REPORT



The screenshot shows the Salesforce LWC Editor interface. On the left, the component tree displays a folder named 'toastMessage' containing 'toastMessage.html', 'toastMessage.js', and 'toastMessage.js-meta.xml'. The right pane shows the 'toastMessage.js' file with the following code:

```
1 import { LightningElement } from 'lwc';
2 import { ShowToastEvent } from 'lightning/platformShowToastEvent';
3
4 export default class ToastMessage extends LightningElement {
5
6     phvVal;
7     handleChange(event){
8         this.phvVal=event.target.value;
9     }
10
11    handleClick(event){
12        if(true){
13            this.showInfoToast();
14        }
15        if(this.phvVal.length==10){
16            console.log("Inside handle click");
17            this.showSuccessToast();
18            this.showWarningToast();
19        }
20        else if(this.phvVal.length<10){
21            console.log("Error condition")
22            this.showErrorToast();
23        }
24    }
25
26    showWarningToast(){
27        const evt=new ShowToastEvent({
28            title:'warning',
29            message:'Phone Number must be of 10 digit.',
30            variant:'warning',
31            mode:'dismissible'
32        });
33        console.log('About to dispatch event');
34        this.dispatchEvent(evt);
35    }
36
37    showErrorToast(){
38        const evt=new ShowToastEvent({
39            title:'Error',
40            message:'Entered phone number is less than 10.',
41            variant:'error',
42            mode:'dismissible'
43        });
44        this.dispatchEvent(evt);
45    }
46
47    showInfoToast(){
48        const evt=new ShowToastEvent({
49            title:'Info',
50            message:'Phone number should be of 10 digit.',
51            variant:'info',
52            mode:'dismissible'
53        });
54        console.log('About to dispatch event');
55        this.dispatchEvent(evt);
56    }
57
58    showSuccessToast(){
59        const evt=new ShowToastEvent({
60            title:'Success',
61            message:'Phone number is correct',
62            variant:'success',
63            mode:'dismissible'
64        });
65        console.log('About to dispatch event');
66        this.dispatchEvent(evt);
67    }
68 }
```



The screenshot shows the Salesforce LWC Editor interface, similar to the first one but with a tooltip visible over the 'evt' variable in the 'showErrorToast()' method. The tooltip contains the following information:

evt -> ShowToastEvent  
A ShowToastEvent is an event object that can be used to display a toast message to the user. It has the following properties:  
title -> string  
The title of the toast message.  
message -> string  
The message displayed in the toast.  
variant -> string  
The variant of the toast message, such as 'error', 'info', or 'success'.  
mode -> string  
The mode of the toast message, such as 'dismissible' or 'sticky'.

The rest of the code in the 'toastMessage.js' file is identical to the first screenshot.