



Capstone Project Report On “Travel App”

Submitted by:

Name: SAGAR KUMAR GUPTA

Batch Wipro Salesforce B13 (WIP-SF-13)

LMS Id: MGSA_490

Date: 28/02/2023

Table of contents

S.no	Content	Page No.
1.	Abstract	1
2.	Introduction	2
3.	Flow of the project	3
4.	Software Requirements	4
5.	Screen shots	
	5(a). Module - 1	5-20
	5(b). Module - 2	21-28
	5(c). Module - 3	29-56
6.	References	57

Abstract

Salesforce, Inc. is an American cloud-based software company headquartered in San Francisco, California. It provides customer relationship management (CRM) software and applications focused on sales, customer service, marketing automation, analytics, and application development.

Salesforce's main technologies are tools for customer management. Other products enable customers to create apps, integrate data from other systems, visualize data, and offer training courses.

Force.com applications are built using declarative tools, backed by Lightning and Apex, a proprietary Java-like programming language for Force.com, as well as Visualforce, a framework including an XML syntax typically used to generate HTML. The Force.com platform typically receives three complete releases a year. As the platform is provided as a service to its developers, every single development instance also receives all these updates.

In here we work on how to Set up the Company Profile, Configuring the user Interface, setting up Activities and Calendars, Configuring Search Settings, Setting up Chatter Groups, Mobile Access with salesforce.

Introduction

About Salesforce:-

Salesforce is a cloud-based software company that provides its customers with a platform to develop their own applications without following the tough steps that they used to follow in the legacy system. The software or application once created can be uploaded onto the cloud allowing the end-users to view them.

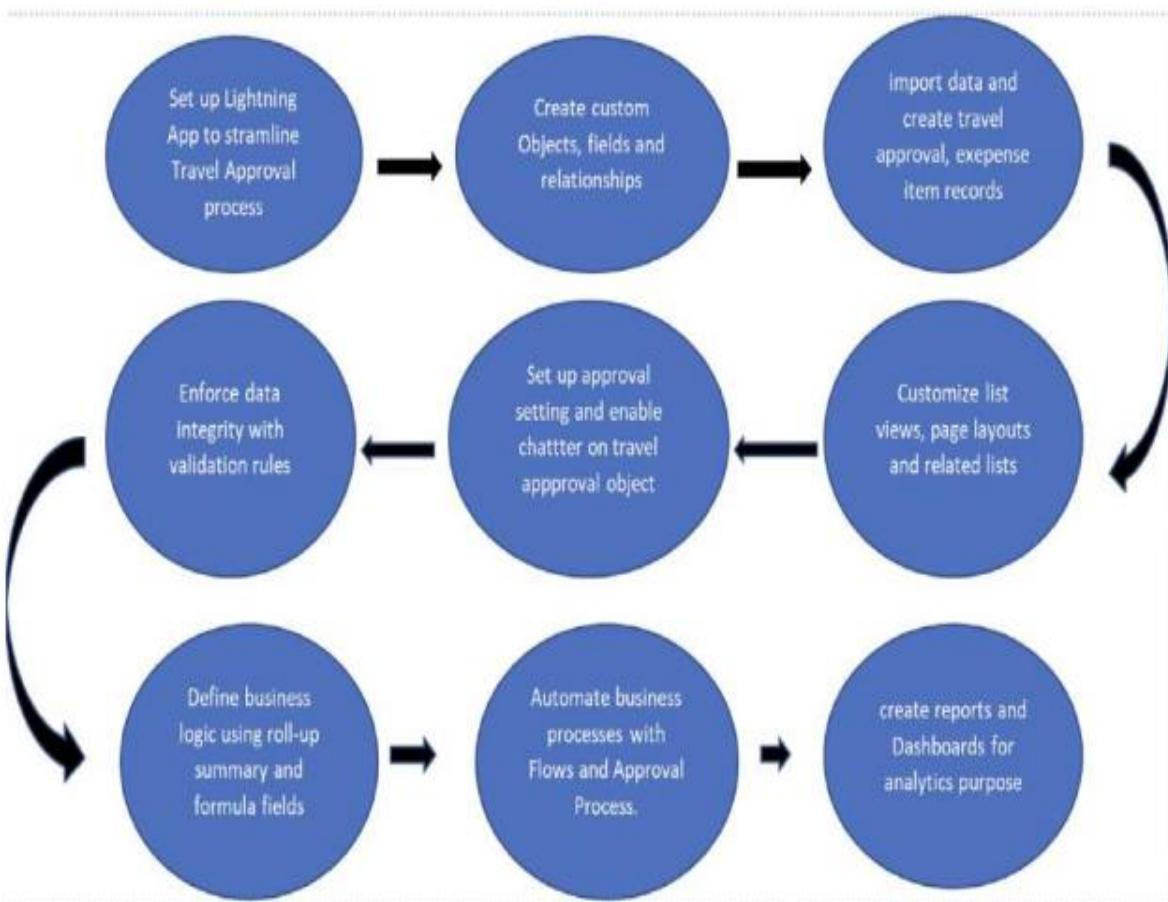
Salesforce is currently providing various software solutions and platforms for developers to create and distribute custom software/applications. Tech giants like Google, Twitter, Amazon, and Facebook are using Salesforce either in the form of SaaS or PaaS.



Salesforce developers can make an application on the cloud and share it with multiple companies across multiple domains by using Salesforce.

Talking about HR systems, every company across the globe has an HR team. Each HR team would require an HR application to store employee records. Almost all specifications for such an application would be common for all companies. So, as a developer, it would be very easy to create a Salesforce application for such specifications, post it onto the cloud, and provide it as a service to multiple clients at the same time. Maintenance of the same can be done altogether too. So basically, the problem of scalability gets eliminated.

Flow of the Project



Software Requirements

For the fastest and most stable experience, we recommend:

- An Octane 2.0 score of 30,000 or greater
- Network latency of 150 ms or less
- Download speed of 3 Mbps or greater
- At least 8 GB of RAM, with 3 GB available for Salesforce browser tabs

Minimum requirements are:

- An Octane 2.0 score of 20,000 or greater
- Network latency of 200 ms or less
- Download speed of 1 Mbps or greater
- At least 5 GB of RAM, with 2 GB available for Salesforce browser tabs

OR

Requirements	
Windows	
Operating system	Windows 8.1 64-bit, Windows 8 64-bit, Windows 7 Service Pack 1 64-bit, Windows Vista Service Pack 2 64-bit
CPU	Core 2 Quad Q6600 at 2.4 GHz or AMD Phenom 9850 at 2.5 GHz
Memory	4 GB RAM
Free space	65 GB of free space
Graphics hardware	DirectX 10-compatible GPU: GeForce 9800GT 1GB or ATI Radeon HD 4870 1GB
Sound hardware	DirectX 10 compatible sound card

Screen shots

Module – 1

Exercise 1: -

Step 1: - Create a new custom lightning App, name: **Travel App**

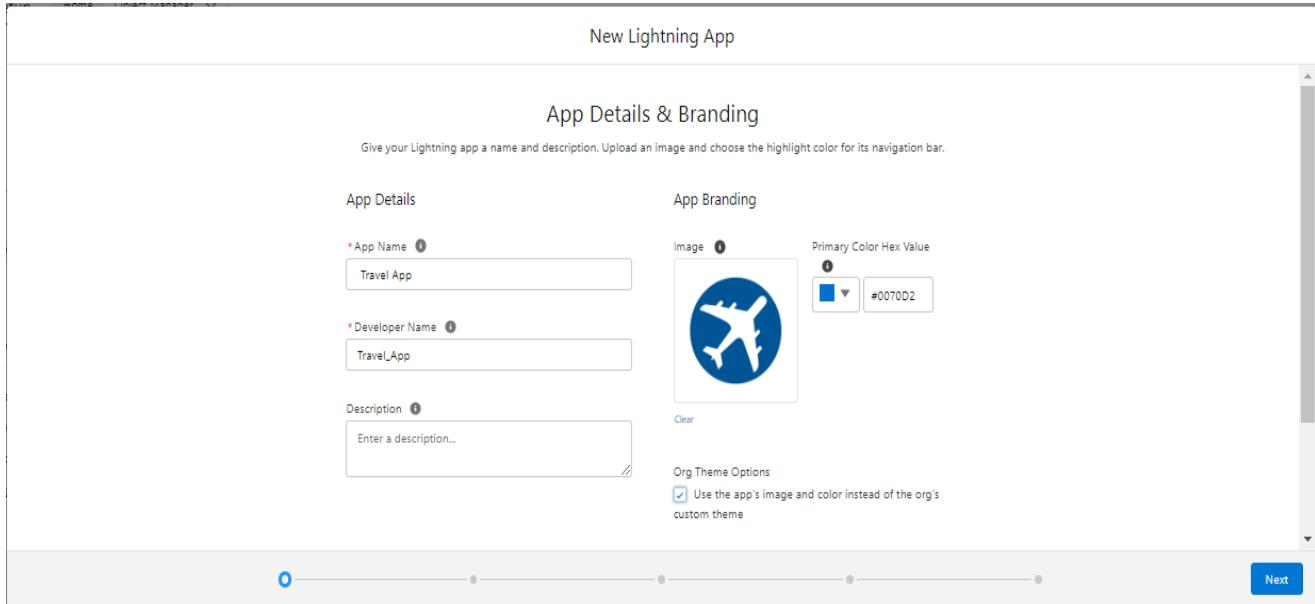
New Lightning App

App Details & Branding

Give your Lightning app a name and description. Upload an image and choose the highlight color for its navigation bar.

App Details	App Branding
* App Name <small>i</small> <input type="text" value="Travel App"/>	Image <small>i</small>  Primary Color Hex Value <input type="color" value="#0070D2"/> #0070D2
* Developer Name <small>i</small> <input type="text" value="Travel_App"/>	Org Theme Options <input checked="" type="checkbox"/> Use the app's image and color instead of the org's custom theme
Description <small>i</small> <input type="text" value="Enter a description..."/>	

Next



New Lightning App

App Options

Navigation and Form Factor i

* Navigation Style
 Standard navigation
 Console navigation

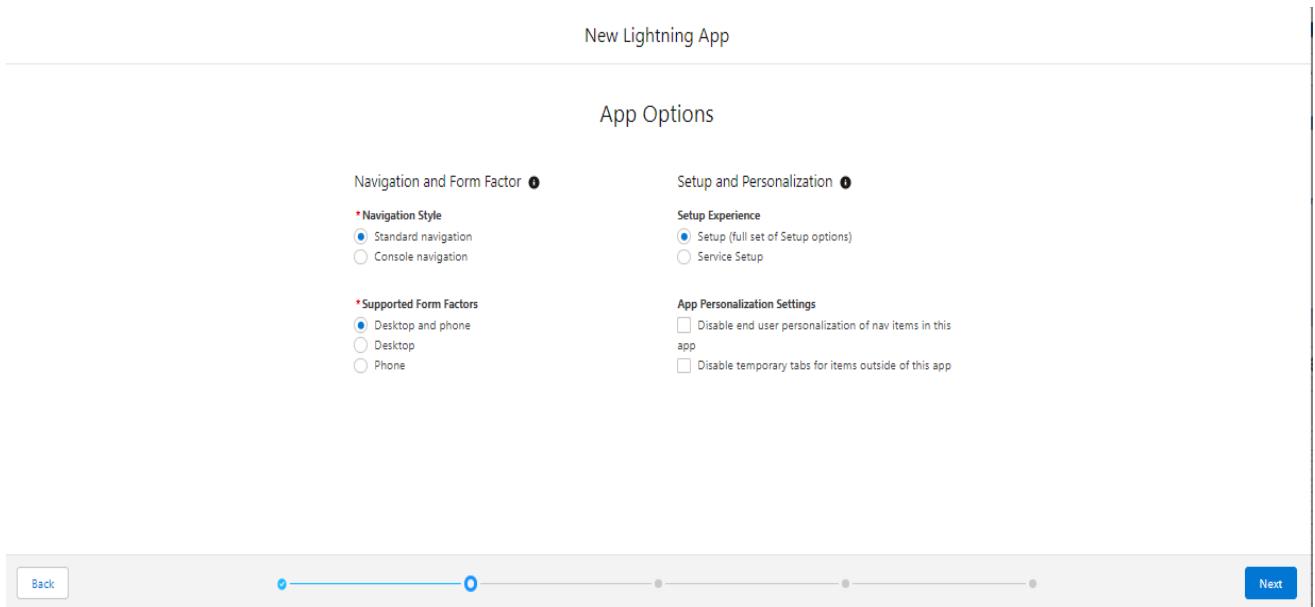
* Supported Form Factors
 Desktop and phone
 Desktop
 Phone

Setup and Personalization i

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

App Personalization Settings
 Disable end user personalization of nav items in this app
 Disable temporary tabs for items outside of this app

Back Next



New Lightning App

Available Items Selected Items

Chatter
Reports
Dashboards

Back Next

New Lightning App

Available Profiles Selected Profiles

System Administrator

Back Save & Finish

Final output of step 1 -

Travel App

Chatter Reports Dashboards

What I Follow

To Me

Bookmarked

Company Highlights

My Drafts

STREAMS +

You don't have any streams yet. Try creating one!

RECENT GROUPS +

Aw, you don't have any groups! Why not create or join some now?

Post Poll Question

Share an update... Share

Sort by: Top Posts

Search this feed...

Sagar Gupta
curious-goat-8quumyo-dev-ed.trailblaze.my.s...
Settings Log Out

DISPLAY DENSITY
✓ Comfy
Compact

OPTIONS
Switch to Salesforce Classic
Add Username

Step 2 : - Create a Department custom object.

Setup->Object Manager->Create Custom object (From top left dropdown button)

The screenshot shows the 'New Custom Object' setup page. In the 'Custom Object Information' section, the 'Label' is set to 'Department' and the 'Plural Label' is set to 'Departments'. Under 'Context-Sensitive Help Setting', the option 'Open the standard Salesforce.com Help & Training window' is selected. In the 'Enter Record Name Label and Format' section, the 'Record Name' is set to 'Department Name' and the 'Data Type' is set to 'Text'. A note states that the Record Name appears in page layouts, key lists, related lists, lookups, and search results.

Step 3 :-Create the following Custom Field in Department Object.

(a). Department Code, Text, Length = 10, Required, Select Unique & Case sensitive

The screenshot shows the 'Department' object's 'Fields & Relationships' tab. A new field is being created with the following details:

- Field Label:** Department Code
- Length:** 10
- Field Name:** Department_Code
- Description:** (empty)
- Help Text:** (empty)
- Required:** checked
- Unique:** checked
- External ID:** unchecked
- Auto add to custom report type:** checked
- Default Value:** (Show Formula Editor) (empty)

(b). Location, Picklist, Value: Kolkata, Delhi.

Setup > Object Manager > Department

Department
New Custom Field

Step 2. Enter the details

Field Label: Location

Values:

- Use global picklist value set
- Enter values, with each value separated by a new line

Kolkata
Delhi

Display values alphabetically, not in the order entered
Use first value as default value
Restrict picklist to the values defined in the value set (checked)

Field Name: Location

Description:

Help Text:

(c). Department Type, Picklist, Values: Banking, Finance, Education, Energy, IT.

Setup > Object Manager > Department

Department
New Custom Field

Step 2. Enter the details

Field Label: Department Type

Values:

- Use global picklist value set
- Enter values, with each value separated by a new line

Banking
Finance
Education
Energy

Display values alphabetically, not in the order entered
Use first value as default value
Restrict picklist to the values defined in the value set (checked)

Field Name: Department_Type

Description:

Help Text:

(d). Create Field Dependency, Controlling field =Location, Dependent field = Department Type

Controlling Field: Location
Dependent Field: Department Type

Instructions

- Double click on a cell to toggle its visibility for the Controlling Field value shown in the column heading.
- To change multiple cells at once, select multiple cells and then click the Include Values or Exclude Values button to change the visibility of all selected cells at once.
- Use SHIFT + click to select a range of adjacent cells. Use CTRL + click to select multiple cells that are not adjacent.
- Use the Preview button to test the results.

Legend

Excluded Value
Included Value

Click button to include or exclude selected values from the dependent picklist:

Showing Columns: 1 - 2 (of 2) < Previous | Next > [View All](#) [Go to](#)

Location:	Kolkata	Delhi
Department Type:	Banking	Banking
	Finance	Finance
	Education	Education
	Energy	Energy
	IT	IT

Click button to include or exclude selected values from the dependent picklist:

Showing Columns: 1 - 2 (of 2) < Previous | Next > [View All](#)

Step 4 : - Create a Travel Approval custom object.

Setup->Object Manager->Create Custom object (From top left dropdown button)

SETUP

New Custom Object

Custom Object Definition Edit

Save Save & New Cancel

Custom Object Information

The singular and plural labels are used in tabs, page layouts, and reports.

Label: Example: Account
 Plural Label: Example: Accounts
 Starts with vowel sound:

The Object Name is used when referencing the object via the API.

Object Name: Example: Account

Description:

Context-Sensitive Help Setting

Open the standard Salesforce.com Help & Training window
 Open a window using a Visualforce page

Content Name:

Enter Record Name Label and Format

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name: Example: Account Name
 Data Type:

Now, Create the following Custom Field in Travel Approval Object.

- Purpose of Trip, Text Area.
- Status, Picklist, Values = New, Submitted, Pending Approval, Approved, Rejected, Draft.
- Trip Start Date, Date.
- Trip End Date, Date.
- Out of State, Checkbox.
- Destination State, Text, Length = 2.
- Department, Lookup, Related to = Department custom object.
- Save

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD
Created By	CreatedBy	Lookup(User)	
Department	Department_c	Lookup(Department)	
Destination State	Destination_State_c	Text(2)	
Last Modified By	LastModifiedBy	Lookup(User)	
Out Of State	Out_Of_State_c	Checkbox	
Owner	OwnerId	Lookup(User/Group)	
Purpose Of Trip	Purpose_of_Trip_c	Text Area(255)	
Status	Status_c	Picklist	
Status Indicator	Status_Indicator_c	Formula (Text)	
Total Expenses	Total_Expenses_c	Roll-Up Summary (SUM Expense Item)	
Travel Approval #	Name	Auto Number	
Trip End Date	Trip_End_Date_c	Date	

Step 5 : - Import the data store in Department.csv by using “Data Import wizard” tool.

5(a) -

5(b) -

Edit	Mapped Salesforce Object	CSV Header	Example	Example	Example
Change	Department Name	Department Name	Audit Services	Contract Management	Division of Finance
Change	Department Code	Department Code	405-01	405-02	405-03

-Test the App, till yet what we have done.

1	Audit Services
2	Contract Management
3	Disability Determination Bureau
4	Division of Aging
5	Division of Disability and Rehabilitative Services
6	Division of Family Resources
7	Division of Finance
8	Division of Mental Health and Addiction
9	Human Resources
10	Legislative Services
11	Office of Communications and Media
12	Office of Early Childhood and Out-of-School Learning
13	Office of General Counsel
14	Office of Medicaid Policy and Planning
15	Quality and Compliance Office
16	Technology

See the data is imported Successfully.

Note :- Before Importing the data make sure you clean the data otherwise some data will not be imported successfully.

Exercise 2: -

Step 1: - Create Travel Approval Record.

The screenshot shows a Salesforce page for a 'Travel Approval' record with ID TA00001. The page has a 'Details' tab selected. Key fields visible include:

- Travel Approval #: TA00001
- Purpose Of Trip: Attend Dreamforce
- Status: Draft
- Trip Start Date: 28/02/2023
- Trip End Date: 05/03/2023
- Out Of State:
- Destination State: CA
- Department: Technology
- Created By: Sagar Gupta, 23/02/2023, 11:49 am
- Last Modified By: Sagar Gupta, 23/02/2023, 11:49 am

The right side of the page displays an 'Activity' section with a toolbar for filtering and viewing activities. It shows a message: "No activities to show. Get started by sending an email, scheduling a task, and more." Below this, it says "No past activity. Past meetings and tasks marked as done show up here."

Step 2 : - Create a Expense Items custom object.

Setup->Object Manager->Create Custom object (From top left dropdown button)

The screenshot shows the 'Object Manager' setup screen. A new object named 'Expense Item' is being created. The 'Details' tab is selected, showing the following configuration:

- Description: (empty)
- API Name: Expense_Item__c
- Custom:
- Singular Label: Expense Item
- Plural Label: Expense Items
- Enable Reports:
- Track Activities:
- Track Field History:
- Deployment Status: Deployed
- Help Settings: Standard salesforce.com Help Window

The left sidebar lists various object settings: Fields & Relationships, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Restriction Rules, Scoping Rules, Triggers, Flow Triggers, and Validation Rules.

Step 3 :-Create the following custom fields on Expense Item Object.

- Amount, Length = 16, Type = Currency, Decimal = 2, Required = True.
- Expense Type, Type = Picklist, Values = Airfare, Hotel, Rental Car, Meals, Other, Required = True.
- Travel Approval, Type = Master-Detail, Related To – Travel Approval.

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

- Setup:** The top navigation bar includes links for Home, Object Manager, and a search bar labeled "Search Setup".
- Object Manager:** The main title is "Expense Item".
- Fields & Relationships:** This section is currently selected. It displays a table of fields:

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Amount	Amount__c	Currency(16, 2)		
Created By	CreatedById	Lookup(User)		
Expense Item Number	Name	Auto Number	✓	
Expense Type	Expense_Type__c	Picklist		
Last Modified By	LastModifiedById	Lookup(User)		
Travel Approval	Travel_Approval__c	Master-Detail(Travel Approval)	✓	
- Left sidebar:** A vertical list of setup categories including 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, Scoping Rules, Triggers, Flow Triggers, and Validation Rules.

Step 4 :-Create the Expense Items .

- Amount = 870, Expense Type = “Hotel”.
- Save.
- Amount = “450”, Expense Type = “Airfare”
- Save.

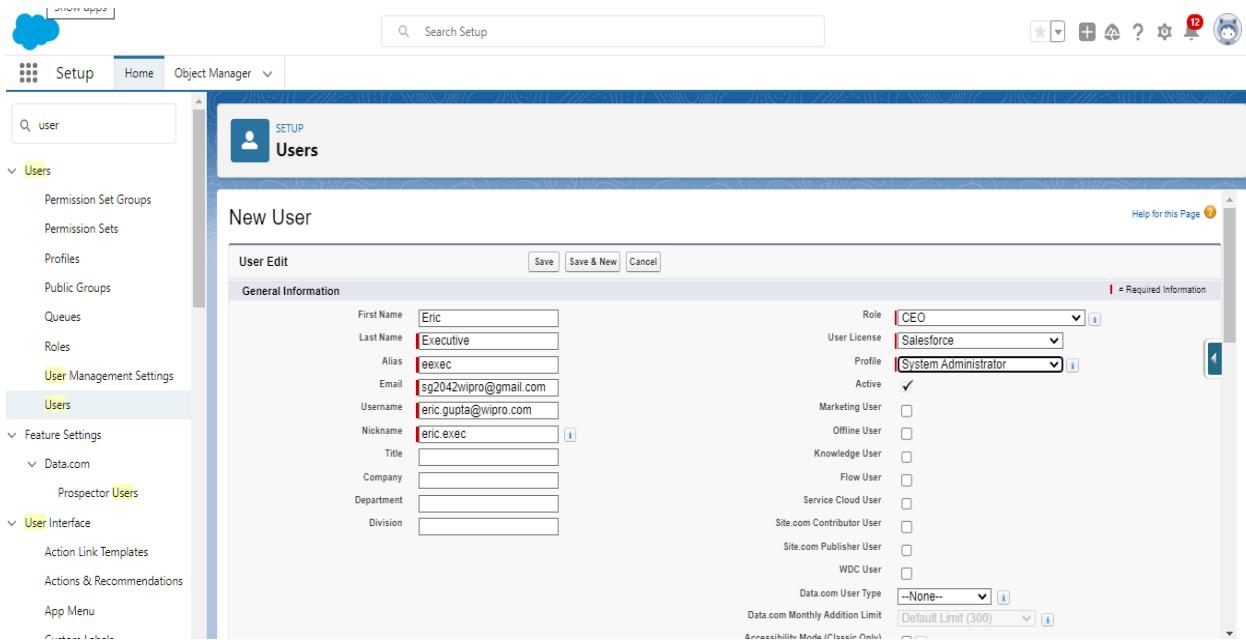
The screenshot shows the Travel App interface with the following details:

- Travel App:** The top navigation bar includes links for Chatter, Reports, Dashboards, Departments, and Travel Approvals.
- Travel Approval:** The main title is "TA00001".
- Related:** This section shows a list of Expense Items (2):

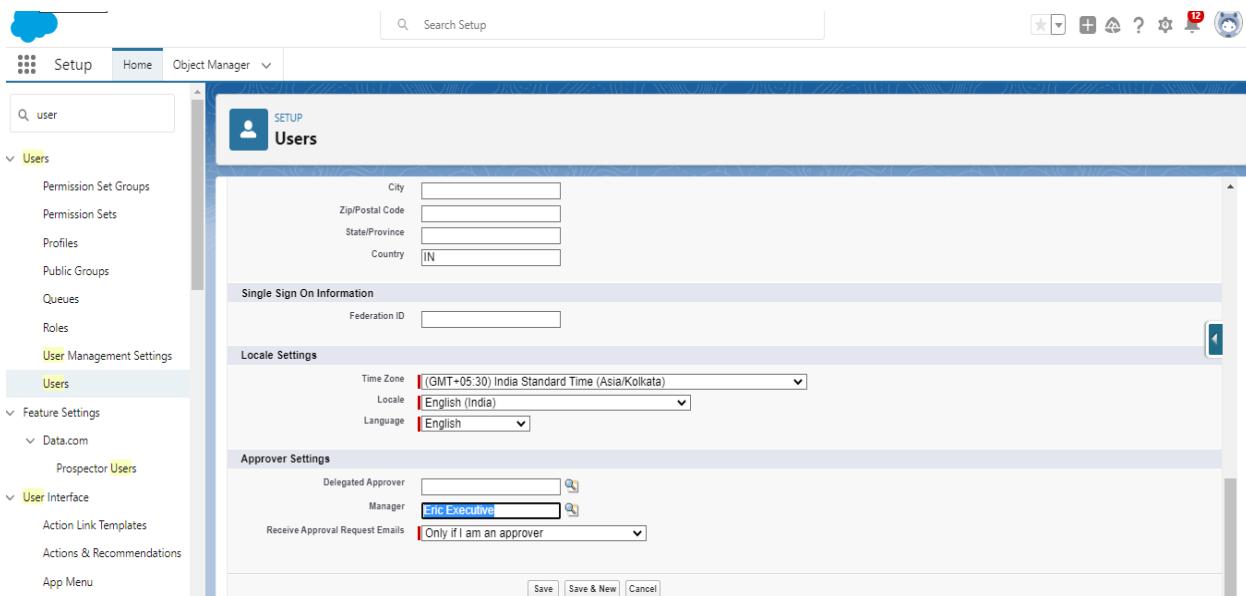
Expense Item Number
E-00001
E-00002
- Activity:** This section displays activity history and scheduled tasks. It includes buttons for New Contact, Edit, and New Opportunity.
- Activity sidebar:** Shows filters for All time, All activities, All types, and options to Refresh, Expand All, and View All. It also indicates "No activities to show" and "Get started by sending an email, scheduling a task, and more."

Step 5 :- Create a user.

- First Name = “Eric”, Last Name = “Executive”, Email = “Use your own email”,
- Username Name = “Choose a Unique username”
- Role = “CEO”
- License = Salesforce.
- Profile = System Administrator.
- Save.



Step 6:- Add user Eric Executive as your manager as shown in the screen shot.



Step 7:- Customize the Travel Approval Default search layout as shown in the screen shot & Save.

Setup > Object Manager

Travel Approval

Edit Search Layout

Travel Approval Search Results

Select the fields to include in this search layout. Note that your choices only determine the display of search results and do not affect the fields that are actually searched. Your selections also determine the fields that are available to users when customizing their search results columns. Please refer to the online help for [more information on search fields](#).

Available Fields	Selected Fields
Record ID	Travel Approval #
Out Of State	Purpose Of Trip
Owner Alias	Department
Owner First Name	Status
Owner Last Name	Destination State
Created By Alias	Trip Start Date
Created By	Trip End Date
Created Date	
Last Modified By Alias	
Last Modified By	
Last Modified Date	

Override the search result column customizations for all users

Standard Buttons
There are no customizable standard buttons for this view.

Custom Buttons
Customize the search results columns

Step 8:- Select fields to display in the Travel Approval “All” List view, as shown in the screen shot & Save.

Travel App

Travel Approvals All

1 item • Sorted by Travel Approval # • Filtered by All travel approvals • Updated

Travel Approval # ↑

1 TA00001

Search...

New Import Change Owner Printable View

Select Fields to Display

Available Fields	Visible Fields
Out Of State	Travel Approval #
Owner Alias	Department
Owner First Name	Created By
Owner Last Name	Status
Purpose Of Trip	Trip Start Date
Record ID	Trip End Date

Cancel Save

Step 9 :- Create Travel approval custom List View “Open Out of State Travel Requests” as shown in the screen shot & Save.

- All users should be able to see this list view.

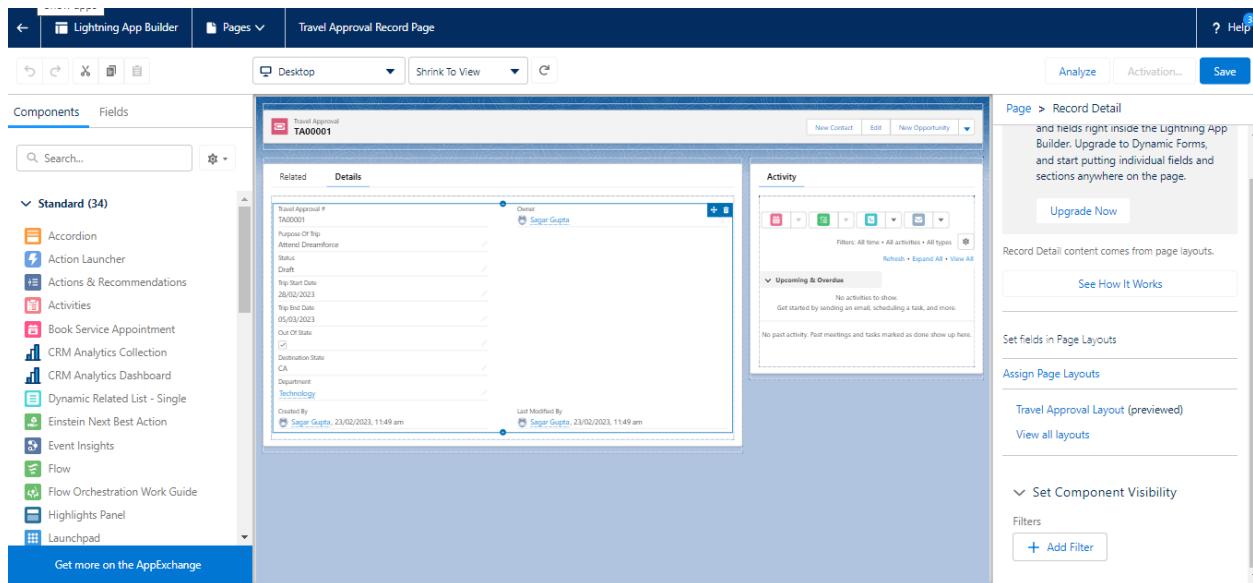
The screenshot shows the Salesforce interface with the 'Travel Approvals' tab selected. A list view titled 'Open Out of State Travel Requests' is displayed, showing one item: TA00001. To the right of the list, a filter sidebar is open, showing two filters applied: 'Out Of State' equals True and Status not equal to Approved, Rejected.

Step 10 :- Select fields to display in the Travel Approval “Open Out of State Travel Requests” List view, as shown in the screen shot & Save.

The screenshot shows the 'Select Fields to Display' dialog box overlaid on the list view from Step 9. The 'Available Fields' list on the left includes Owner Alias, Owner First Name, Owner Last Name, Purpose Of Trip, Record ID, and Travel Approval #. The 'Visible Fields' list on the right includes Department, Created By, Status, Destination State, Trip Start Date, and Trip End Date. The 'Filters' sidebar on the right remains the same as in Step 9.

Step 11 :- (a) Customize the Travel Approval Page Layout as shown in the screen shot.

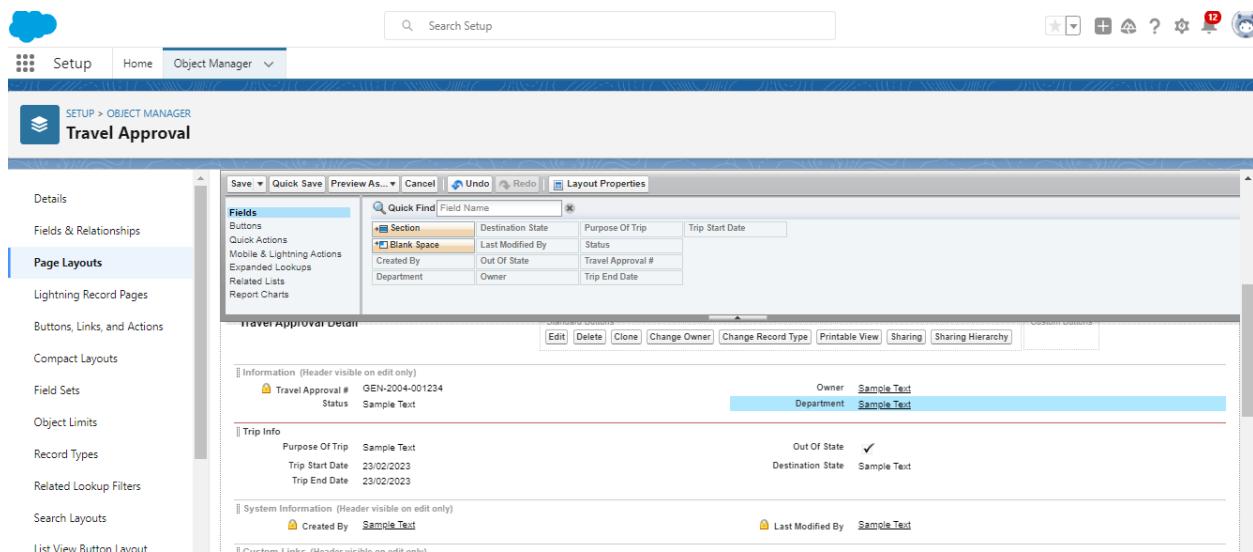
- In the travel app, click travel approval tab and open TA-00001
- Then click Gear button and select “Edit Button”
- It will open as shown as screenshots.



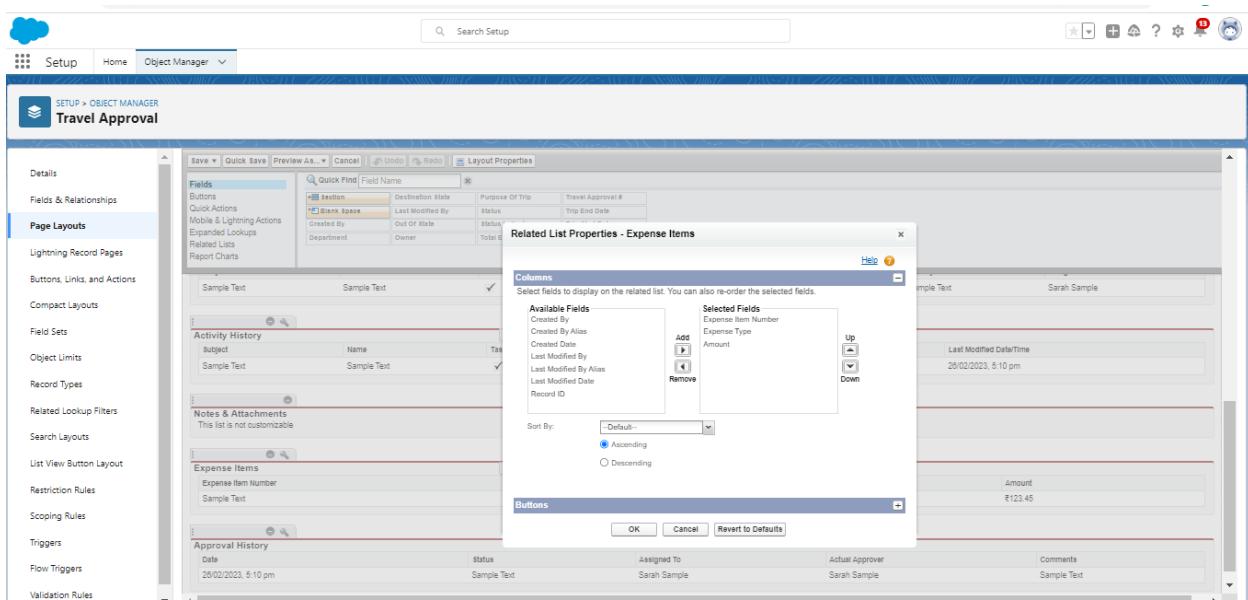
(b) :- Travel Approval Page Layout should look as shown in the screen shot.

Note: Add a section to the page layout called “Trip Info” and add the fields as shown.

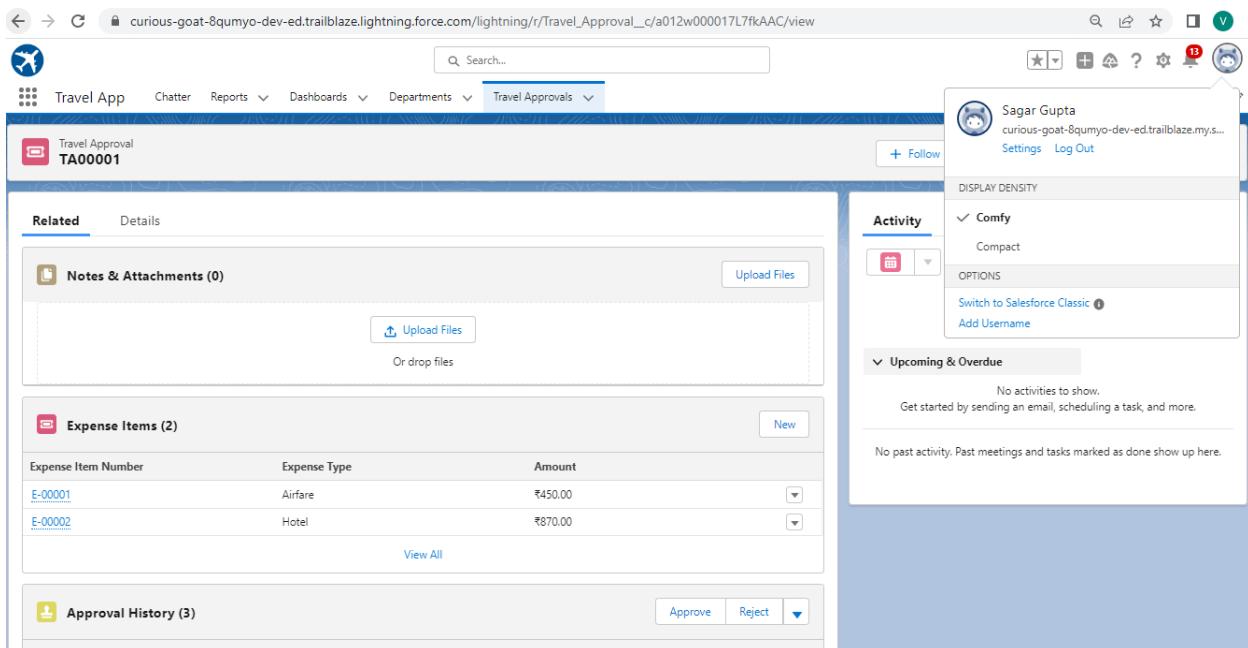
Click Save.



Step 12 :- (a) Customize the Expense Item Related List under the Travel Approval page layout as shown in the screen shots.



(b)- The Travel Approval App should look as shown in the screenshot.



Step 13 :- (a)

- Enable Chatter on the Travel Approval Object.
- Enable “Feed Tracking” for Travel ApprovalObject.
- Select these 2 fields:
- Destination State
- Status
- Save.

The screenshot shows the Salesforce Setup interface with the 'Feed Tracking' page open. In the left sidebar, under 'Chatter', 'Feed Tracking' is selected. On the right, the 'Fields in travel approvals' section is displayed, allowing users to select fields for feed tracking. The 'Travel Approval' object is selected, and its fields are listed: Department, Out Of State, Purpose Of Trip, Travel Approval #, Trip Start Date, Destination State, Owner, Status, and Trip End Date. The 'Enable Feed Tracking' checkbox is checked. A note at the bottom says, "You can also display feed activity for related objects." A 'Save' button is visible at the top right.

(b) Test Collaboration :-

- Open a Travel Approval record.
- Click on Chatter Tab.
- Share a post: Which Department should I associate this travel request with?
- Mention user Eric Executive on the Post using @.
- Note: Login in as Eric and reply to the email, saying: “Technology is the correct department”.
- Note: Enable “Administrator can Log in as any user.”

The screenshot shows the Salesforce Chatter interface. On the left, the 'Travel App' navigation bar is visible. The main area shows a post from 'TA00001 — Sagar Gupta' 9m ago, asking, "@Eric Executive Which department should I associate this travel request with?". Below the post, there is a comment from 'Eric Executive' a few seconds ago, stating, "Technology is the correct department.". A reply box is visible at the bottom for writing a comment.

Test the App , Let see how its look like

- The Travel Approval App should look as shown in the screenshot.

The screenshot shows a Salesforce page for a travel approval record. The top navigation bar includes links for Travel App, Chatter, Reports, Dashboards, Departments, and Travel Approvals. The main content area displays a travel approval record with the ID TA00001. The record details include:

- Travel Approval #: TA00001
- Status: Draft
- Trip Info:
 - Purpose Of Trip: Attend Dreamforce
 - Trip Start Date: 28/02/2023
 - Trip End Date: 05/03/2023
- Owner: Sagar Gupta
- Department: Technology
- Out Of State: checked
- Destination State: CA
- Created By: Sagar Gupta, 23/02/2023, 11:49 am
- Last Modified By: Sagar Gupta, 23/02/2023, 11:49 am

On the right side, there is a Chatter feed with the following entries:

- A post from Sagar Gupta (@Eric Executive) asking, "Which department should I associate this travel request with?" with 1 comment and 1 view.
- A reply from Eric Executive (@Eric Executive) saying, "Technology is the correct department." with 1 like.

At the bottom of the Chatter feed, there is a text input field for writing a comment.

After completing the exercise 2 you will see the user can post their query by using chatter and other can reply to that query.

Now, Here Our module 1 is completed lets move to our next module 2.

Module – 2

Exercise 1: -

Step 1:- Create Validation Rule.

Business Logic: Trip end date must always be greater than (\geq) the trip start date.

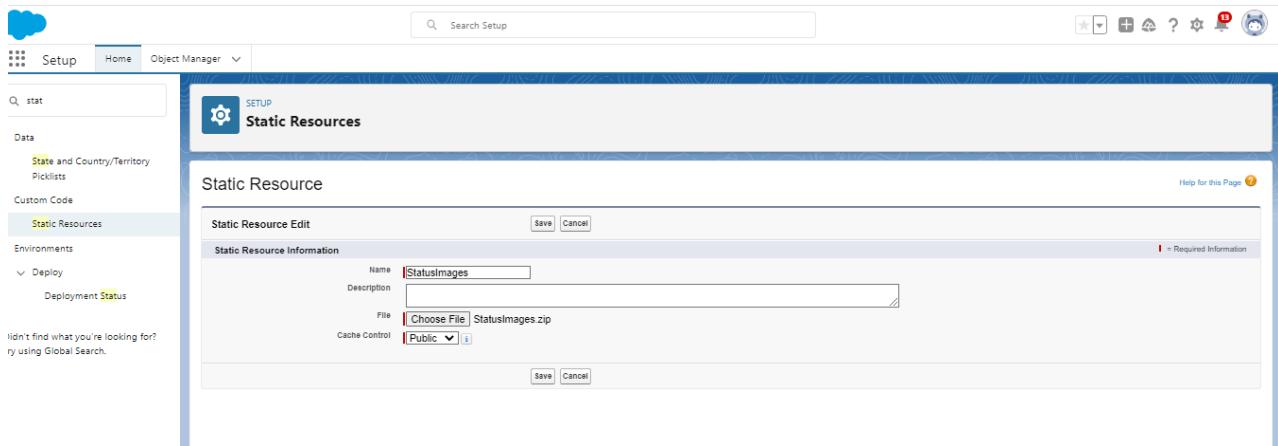
The screenshot shows the 'Validation Rule' section of the Salesforce Object Manager. The validation rule formula is set to `Trip_End_Date__c < Trip_Start_Date__c`. The error message is defined as 'Trip end date must be greater than or equal to start date'. The formula editor includes a dropdown for functions like ABS, ACOS, ADDMONTHS, AND, ASCII, ASIN, and Insert Selected Function. A sidebar on the right provides information about restoring records from the Recycle Bin.

Step 2 :- Create a Roll-Up Summary Field on Travel Approval object.

The screenshot shows the 'New Custom Field' step 3 of 5 for creating a roll-up summary field. The master object is 'Travel Approval' and the summarized object is 'Expense Items'. The roll-up type is set to 'SUM' and the field to aggregate is 'Amount'. The filter criteria is set to 'All records should be included in the calculation'. Navigation buttons for Previous, Next, and Cancel are visible at the bottom.

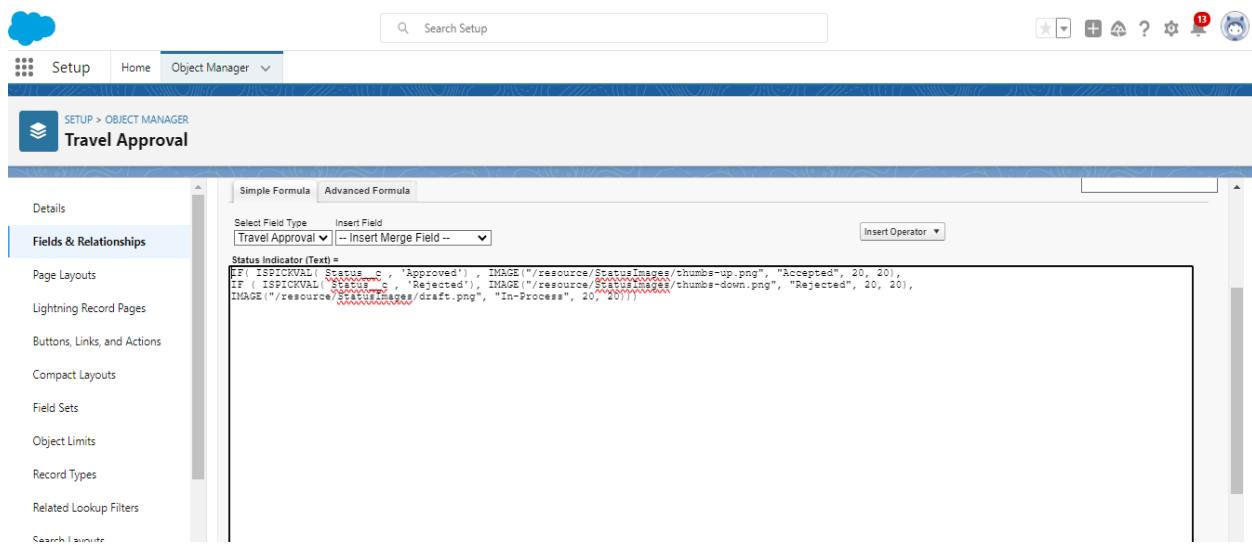
Step 3 :- Business Logic: Create a field that shows a visual indicator based on the value of the Status field

- Setup | Custom Code | Static Resource | New



Step 4 :- Create a Formula field on the Travel Approval object to show an image based on the Status field.

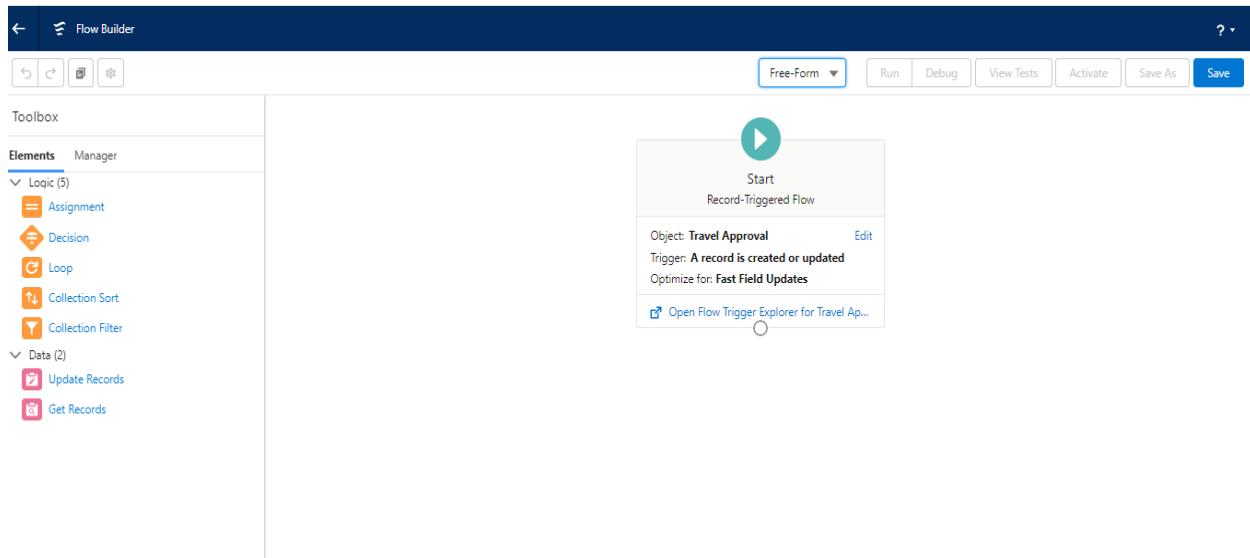
- Field Label: Status Indicator
- Formula Return Type = Text
- Formula: IF(ISPICKVAL(Status c , 'Approved') , IMAGE("/resource/StatusImages/thumbs-up.png", "Accepted", 20, 20), IF (ISPICKVAL(Status c , 'Rejected') , IMAGE("/resource/StatusImages/thumbs-down.png", "Rejected", 20, 20), IMAGE("/resource/StatusImages/draft.png", "In-Process", 20, 20)))
- Save



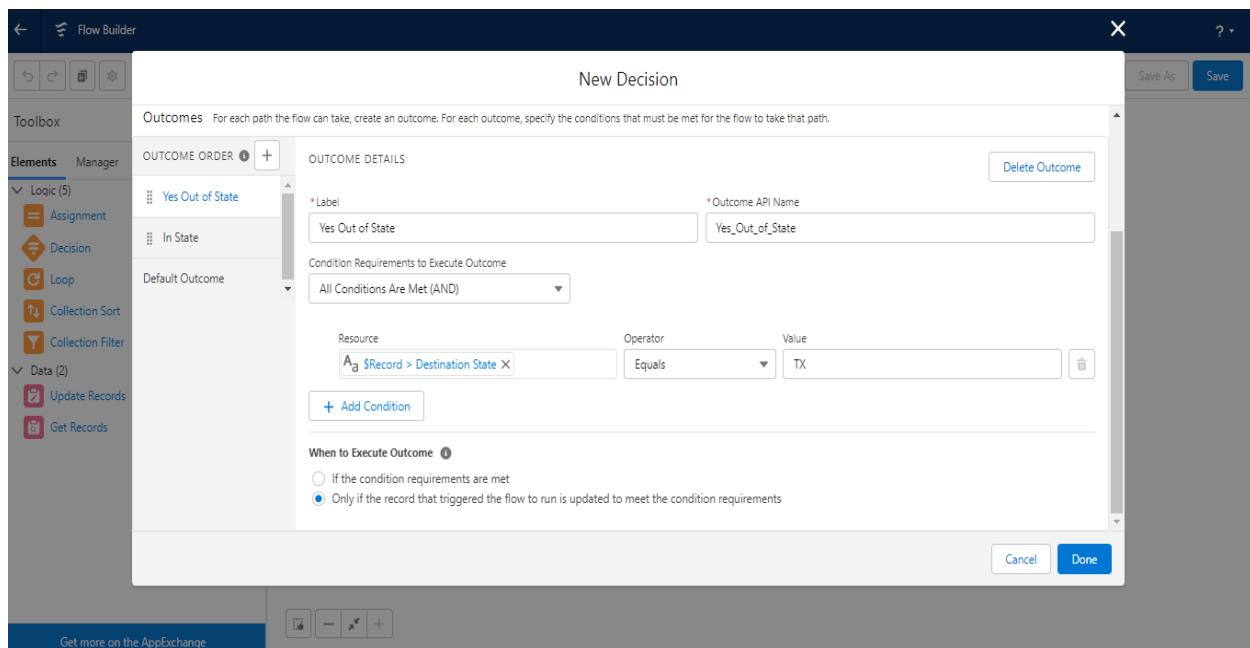
Step 5 :-

(a). Create a Record – Triggered Flow.

- Flow should look like this:

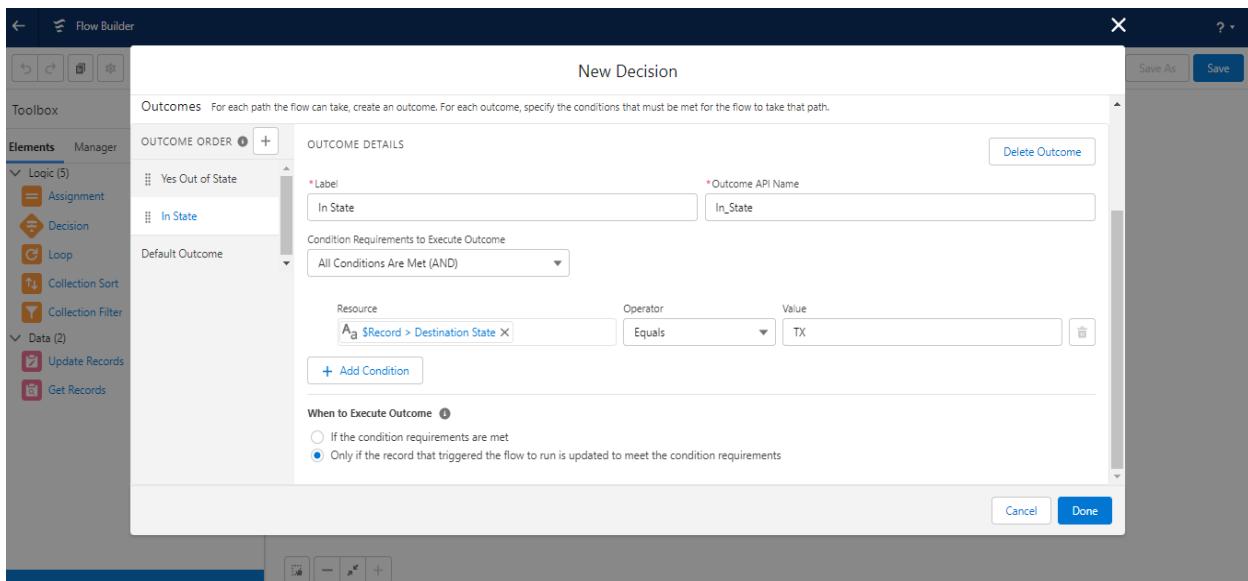


(b). Add a Decision Element to the Flow.

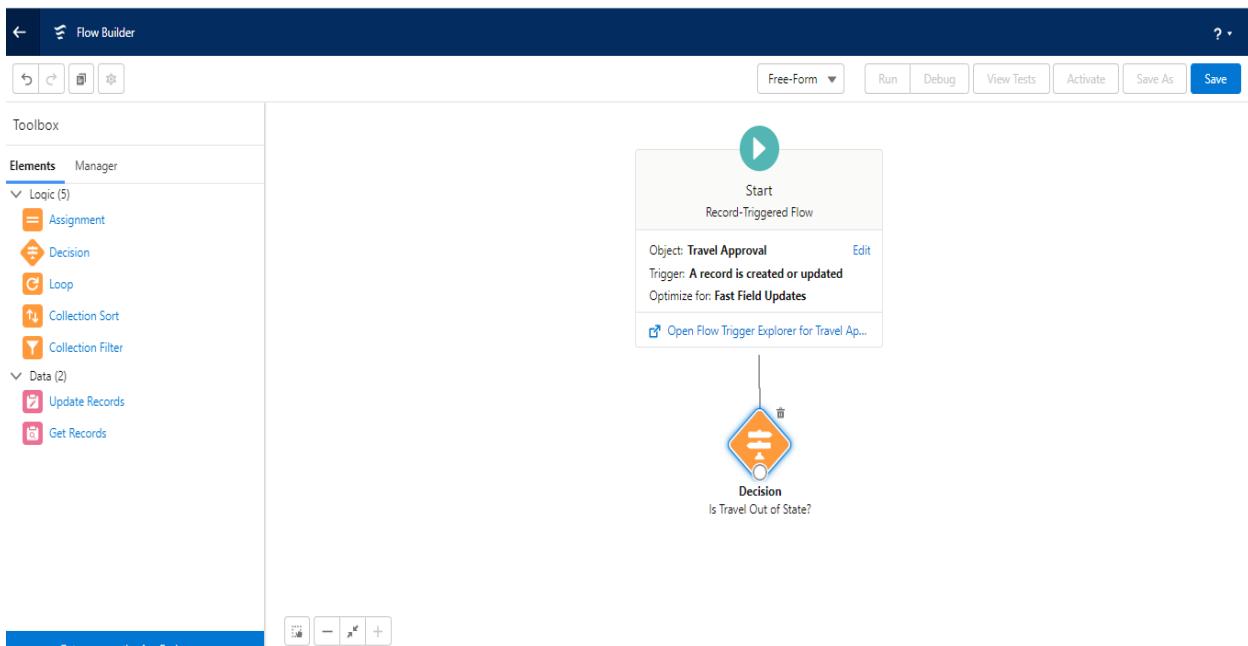


(c). Next to Outcome Order click the + button to add another outcome.

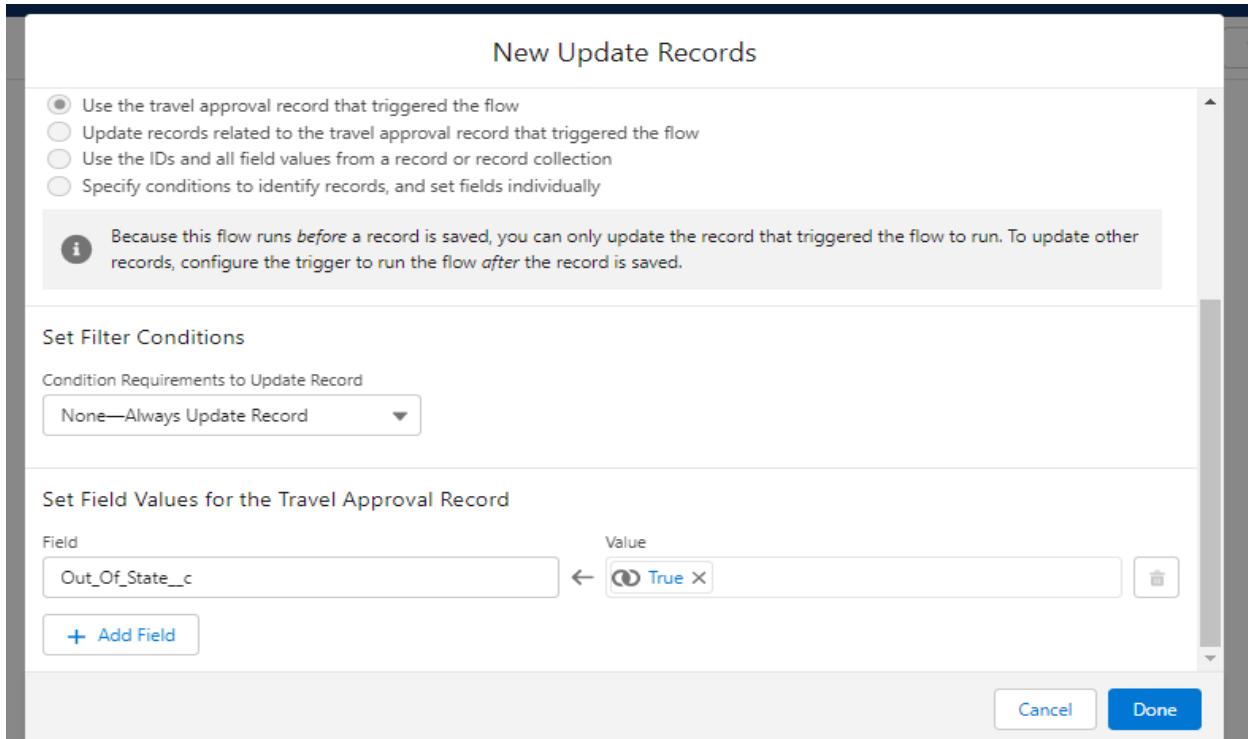
- Flow should look like this:



(d). Connect the Start Flow element to the Decision element.

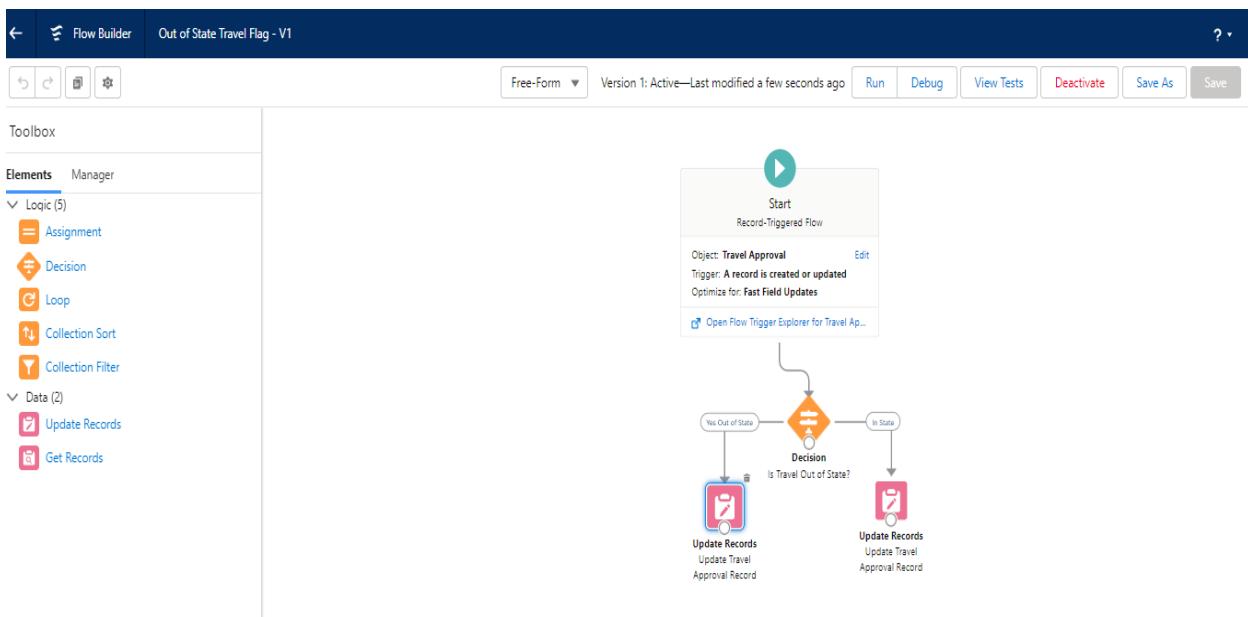


(e). Create an Action for the Flow Using Update Records Elements.



(f). Test The Flow Screen

- Drag the white circle from the Decision Node to the Update Records you just created and select the decision In State | Done.
- Flow screen should look this:



Step 6 :-

- (a). Create an Approval Process to send Travel approvals to Manager or Travel coordinator.

The screenshot shows the Salesforce Setup interface with the 'Approval Processes' page selected. The page title is 'Travel Approval: Travel Approval Request'. The 'Process Definition Detail' section includes fields like Unique Name (Travel_Approval_Request), Description (Travel Approval: Total Expenses GREATER THAN 0), and Record Editability (Administrator ONLY). The 'Initial Submission Actions' section contains an action named 'Record Lock' with a description 'Lock the record from being edited'. The 'Approval Steps' section is currently empty. The page also shows standard Salesforce navigation and search bars.

(b). Test The Approval Process

- Create few Travel Approval records and Submit for Approval.
- Login as Eric, approve and reject the records randomly as shown in the screen shot:

The screenshot shows the Travel App interface with a travel approval record for 'TA00001 Approval'. The record details are: Submitter (Sagar Gupta), Date Submitted (23-Feb-2023), Actual Approver (Eric Executive), and Assigned To (Eric Executive). The status is 'Pending'. On the right, there is a 'Details' section showing Approval Details (Travel Approval # TA00001, Owner Sagar Gupta) and an 'Approver Comments' section where Eric Executive has commented 'Look good. Have fun!' at 23-Feb-2023, 4:43:09 pm.

Exercise 2: -

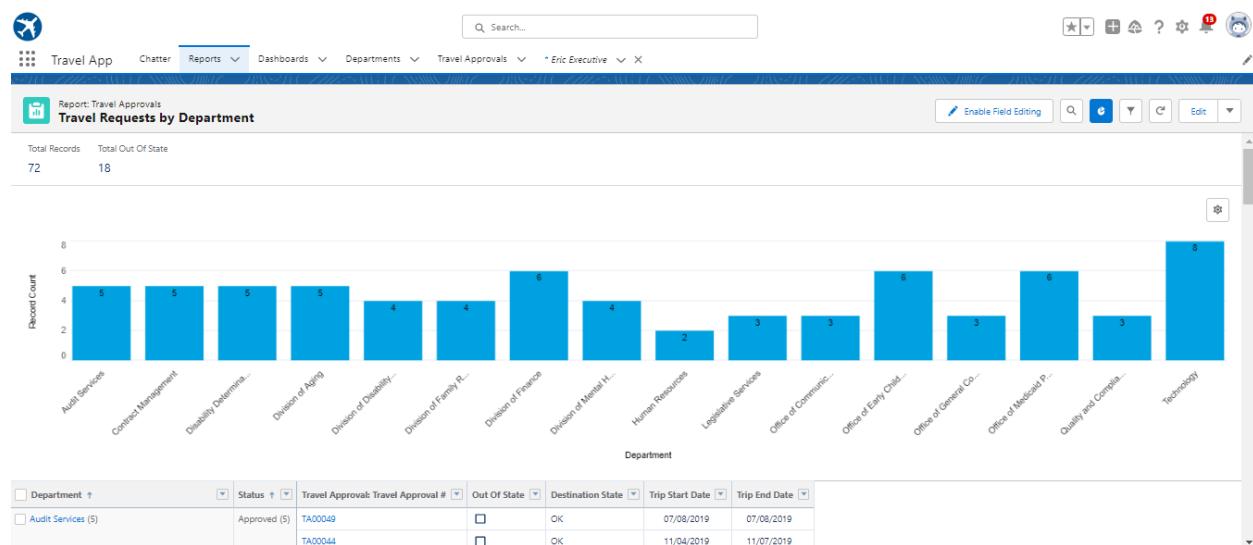
Step 1 :- Use Data Import Wizard to import Travel Approval records.

- Test The App
 - Make sure all the Travel Approval records are successfully imported, check the screenshot:

	Travel Approval #	Department	Created By	Status	Trip Start Date	Trip End Date
1	TA00001	Technology	Sagar Gupta	Draft	28/02/2023	05/03/2023
2	TA00002	Disability Determination Bureau	Sagar Gupta	Approved	10/01/2019	10/01/2019
3	TA00003	Division of Disability and Rehabilitative Services	Sagar Gupta	Rejected	04/03/2019	04/06/2019
4	TA00004	Division of Finance	Sagar Gupta	Rejected	03/09/2019	03/09/2019
5	TA00005	Division of Aging	Sagar Gupta	Approved	11/05/2019	11/11/2019
6	TA00006	Technology	Sagar Gupta	Approved	03/06/2019	03/09/2019
7	TA00007	Division of Disability and Rehabilitative Services	Sagar Gupta	Approved	11/06/2019	11/12/2019
8	TA00008	Disability Determination Bureau	Sagar Gupta	Rejected	03/06/2019	03/07/2019
9	TA00009	Contract Management	Sagar Gupta	Approved	05/11/2019	05/11/2019
10	TA00010	Division of Disability and Rehabilitative Services	Sagar Gupta	Approved	07/07/2019	07/07/2019
11	TA00011	Office of Communications and Media	Sagar Gupta	Approved	06/02/2019	06/12/2019
12	TA00012	Office of Medicaid Policy and Planning	Sagar Gupta	Approved	11/05/2019	11/11/2019
13	TA00013	Division of Family Resources	Sagar Gupta	Approved	04/01/2019	04/01/2019
14	TA00014	Human Resources	Sagar Gupta	Approved	07/09/2019	07/11/2019
15	TA00015	Division of Disability and Rehabilitative Services	Sagar Gupta	Approved	10/11/2019	10/12/2019

Step 2 :- Create a Travel Requests by Department Report.

- Test The Report
- Report might look as per the screen shot:



Step 3 :- Create a Travel Requests by Month Report

- Test The Report
- Report might look as per the screen shot:

The screenshot shows a Salesforce report titled "Travel Requests by Month". The report displays travel approvals for three months: January 2019, February 2019, and March 2019. The data is organized by month, with subtotals for each month and a grand total for the year. The columns include Trip End Date, Out Of State, Travel Approval: Travel Approval #, Department, Status, Destination State, and Trip Start Date.

Trip End Date	Out Of State	Travel Approval: Travel Approval #	Department	Status	Destination State	Trip Start Date
January 2019 (4)	(1)	TA00002	Disability Determination Bureau	Approved	OK	10/01/2019
	Subtotal					
	(3)	TA00039	Division of Finance	Approved	TX	10/01/2019
		TA00072	Office of General Counsel	Approved	TX	04/01/2019
		TA00013	Division of Family Resources	Approved	TX	04/01/2019
	Subtotal					
Subtotal						
February 2019 (1)	(1)	TA00062	Division of Family Resources	Approved	FL	12/02/2019
	Subtotal					
Subtotal						
March 2019 (4)	(2)	TA00024	Audit Services	Approved	OK	09/03/2019
		TA00023	Division of Finance	Approved	GA	09/03/2019
	Subtotal					

Step 4 :- Create a Travel Approvals Dashboard.

- Test The Dashboard
- Dashboard will look as per the screen shot:

The screenshot shows a Salesforce dashboard titled "Travel Requests Dashboard". The dashboard includes two charts: "Travel Requests by Department" and "Travel Requests by Month". The "Travel Requests by Department" chart is a horizontal bar chart showing the record count for different departments across three status categories: Approved, Rejected, and Draft. The "Travel Requests by Month" chart is a vertical bar chart showing the record count for each month from January 2019 to March 2023.

Travel Requests by Department

Department	Status	Record Count
Audit Ser...	Approved	5
Contract ...	Approved	5
Disability ...	Approved	3
	Rejected	2
Division o...	Approved	4
	Rejected	1
Division o...	Approved	3
	Rejected	1
Division o...	Approved	2
	Rejected	2

Travel Requests by Month

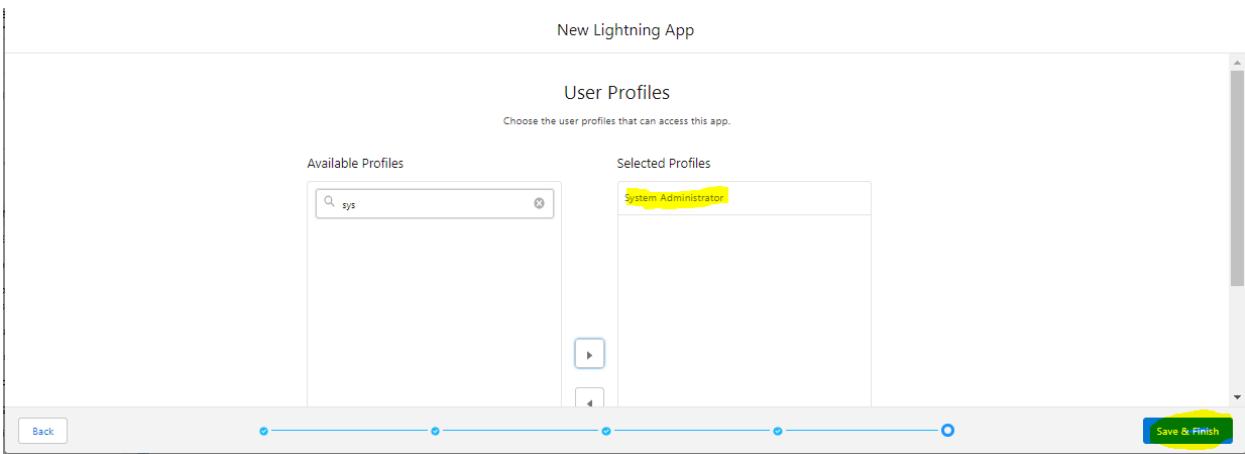
Trip End Date	Record Count
January 2019	4
February 2019	1
March 2019	4
April 2019	10
May 2019	6
June 2019	3
July 2019	8
August 2019	6
September 2019	8
October 2019	7
November 2019	7
December 2019	7
March 2023	1

Module – 3

Exercise 1 : -

Step 1:- Create a new custom lightning App, name: Code Playground.

The screenshot shows the Salesforce Setup interface with the 'Lightning Experience App Manager' selected in the sidebar. The main area displays a table of existing apps, with the 'New Lightning App' button highlighted. The 'App Details & Branding' step is active, showing fields for App Name (Code Playground), Developer Name (Code_Playground), Description (Enter a description...), and App Branding settings (Image, Primary Color Hex Value #0070D2). The 'Org Theme Options' checkbox is checked. The 'Navigation Items' step is shown below, with the 'Available Items' list containing 'case' and the 'Selected Items' list containing 'Accounts', 'Leads', 'Contacts', 'Opportunities', and 'Cases'. Navigation arrows at the bottom indicate the process flow.



Step 2:- Create a Custom Object and tab for Customers.

New Custom Object

Custom Object Definition Edit

Custom Object Information

The singular and plural labels are used in tabs, page layouts, and reports.

Label	Customer	Example: Account
Plural Label	Customers	Example: Accounts
Starts with vowel sound	<input type="checkbox"/>	

The Object Name is used when referencing the object via the API.

Object Name	Customer	Example: Account
-------------	----------	------------------

Description

Tabs

New Custom Object Tab

Step 1. Enter the Details

Choose the custom object for this new custom tab. Fill in other details.

Select an existing custom object or [create a new custom object now](#).

Object	Customer
Tab Style	Credit card

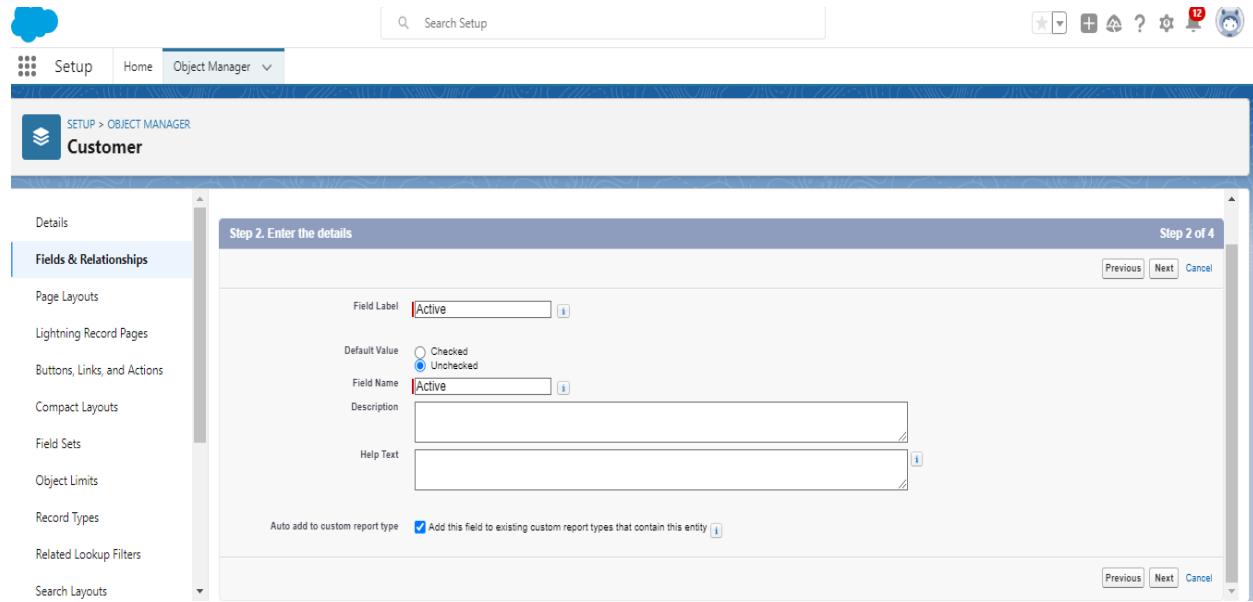
(Optional) Choose a Home Page Custom Link to show as a splash page the first time your users click on this tab.

Splash Page Custom Link: [--None--](#)

Description

Step 3:- Create a Custom fields for Customer Object

a. Label = Active, Checkbox, Save.



Setup > Object Manager > Customer

Step 2. Enter the details

Field Label: Active

Default Value: Checked Unchecked

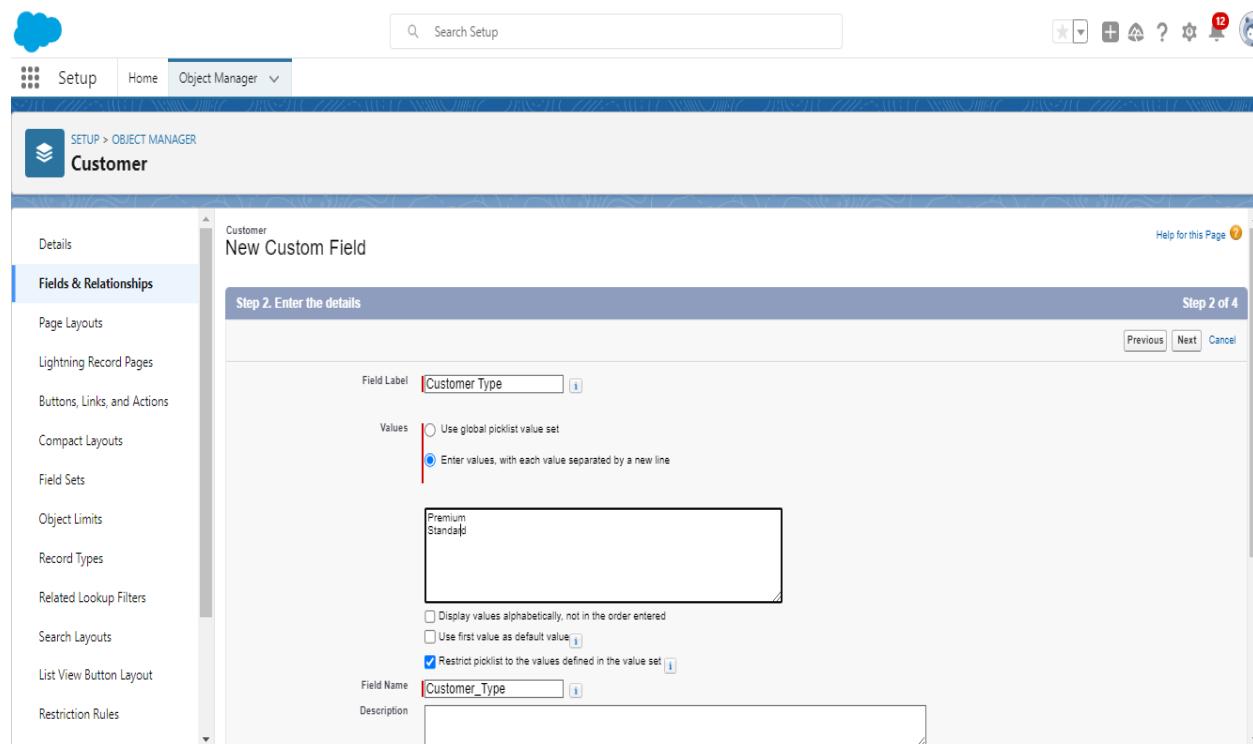
Field Name: Active

Description:

Help Text:

Auto add to custom report type Add this field to existing custom report types that contain this entity

b. Label = Customer Type, Picklist, Values: Premium, Standard



Setup > Object Manager > Customer

New Custom Field

Step 2. Enter the details

Field Label: Customer Type

Values: Use global picklist value set Enter values, with each value separated by a new line

Premium
Standard

Display values alphabetically, not in the order entered
Use first value as default value
 Restrict picklist to the values defined in the value set

Field Name: Customer_Type

Description:

c. Label = Description, Text Area, Save.

The screenshot shows the Salesforce Setup interface for creating a new custom field. The left sidebar is titled 'Customer' and lists various setup options under 'Fields & Relationships'. The main window is titled 'Customer New Custom Field' and is on 'Step 2. Enter the details'. The 'Field Label' is set to 'Description'. The 'Field Name' is also 'Description'. The 'Description' field contains the word 'Description'. The 'Help Text' field is empty. Under 'Required', there is a checkbox 'Always require a value in this field in order to save a record' which is unchecked. Below it is a checked checkbox 'Add this field to existing custom report types that contain this entity'. The 'Default Value' section has a 'Show Formula Editor' button. A note at the bottom explains formula syntax. The top right of the window says 'Step 2 of 4' with 'Previous' and 'Next' buttons.

d. Label = Customer, Master-Detail, Related To – Customer custom object

The screenshot shows the Salesforce Setup interface for creating a new relationship. The left sidebar is titled 'Customer' and lists various setup options under 'Fields & Relationships'. The main window is titled 'Customer New Relationship' and is on 'Step 2. Choose the related object'. The 'Related To' dropdown is set to 'Customer'. The top right of the window says 'Step 2 of 6' with 'Previous' and 'Next' buttons.

Step 4:- Create a Custom Object for Billing.

The screenshot shows the 'New Custom Object' page in the Salesforce Setup interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The main title is 'New Custom Object'. A message bar at the top states: 'Permissions for this object are disabled for all profiles by default. You can enable object permissions in permission sets or by editing custom profiles.' with links to 'Tell me more!' and 'Don't show this message again'. Below this is the 'Custom Object Definition Edit' section with tabs for 'Save', 'Save & New', and 'Cancel'. The 'Custom Object Information' section contains fields for 'Label' (Billing) and 'Plural Label' (Billings), both with examples of 'Account' and 'Accounts'. There is also a checkbox for 'Starts with vowel sound'. The 'Object Name' field is set to 'Billing' with an example of 'Account'. A 'Description' text area is present. Under 'Context-Sensitive Help Setting', there are two radio button options: 'Open the standard Salesforce.com Help & Training window' (selected) and 'Open a window using a Visualforce page'. The 'Content Name' dropdown is set to '--None--'. On the right side of the page, there is a note: 'I = Required Information'.

The screenshot shows the 'New Custom Object Tab' configuration page in the Salesforce Setup interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The main title is 'New Custom Object Tab'. A progress bar at the top indicates 'Step 1 of 3'. The 'Step 1. Enter the Details' section asks to choose a custom object for the new tab. It shows 'Select an existing custom object or [create a new custom object now](#)'. The 'Object' dropdown is set to 'Billing'. The 'Tab Style' dropdown is set to 'Bank' with a magnifying glass icon. Below this, an optional section for a 'Home Page Custom Link' is shown, with the 'Splash Page Custom Link' dropdown set to '--None--'. A 'Description' text area is available for entering a short description. At the bottom right, there are 'Next' and 'Cancel' buttons.

Step 5:- Create a Custom fields for Billing Object.

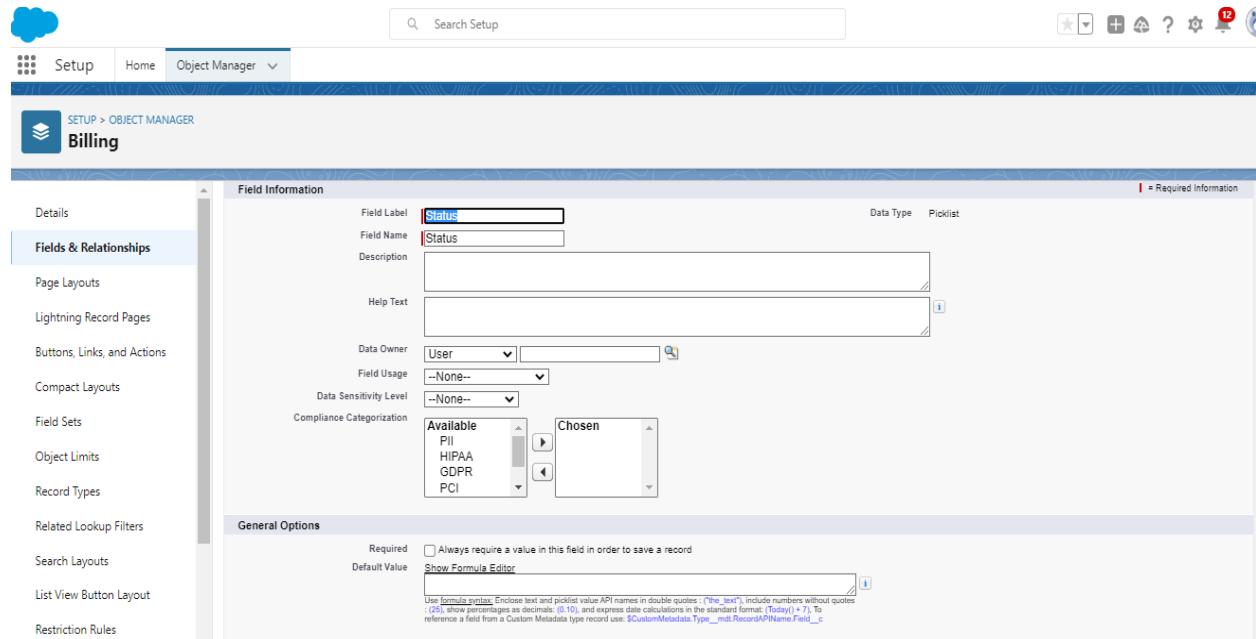
- Label = Amount Paid, Currency, Save.

The screenshot shows the Salesforce setup interface for creating a custom field. The left sidebar is titled 'Fields & Relationships' and includes options like Details, Page Layouts, Lightning Record Pages, Buttons, Links, and Actions, Compact Layouts, Field Sets, Object Limits, Record Types, Related Lookup Filters, Search Layouts, List View Button Layout, and Restriction Rules. The main area is titled 'Edit Billing Custom Field Amount Paid' under 'Custom Field Definition Edit'. The 'Field Information' section shows 'Field Label' as 'Amount Paid', 'Field Name' as 'Amount_Paid', 'Data Type' as 'Currency', and 'Description' and 'Help Text' fields. Under 'Compliance Categorization', 'Available' categories include PII, HIPAA, GDPR, and PCI, while 'Chosen' categories are empty. At the bottom, there's a 'General Options' section with a 'Required' checkbox and a note about always requiring a value.

- Label = Customer Type, Picklist, Values: Premium, Standard.

The screenshot shows the Salesforce setup interface for creating a custom field. The left sidebar is identical to the previous one. The main area is titled 'Edit Billing Custom Field Customer Type' under 'Custom Field Definition Edit'. The 'Field Information' section shows 'Field Label' as 'Customer Type', 'Field Name' as 'Customer_Type', 'Data Type' as 'Picklist', and 'Description' and 'Help Text' fields. Under 'Compliance Categorization', 'Available' categories include PII, HIPAA, GDPR, and PCI, while 'Chosen' categories are empty. At the bottom, there's a 'General Options' section with a 'Required' checkbox and a note about always requiring a value, and a 'Default Value' field containing 'Show Formula Editor'.

c. Label = Status, Picklist, Values: Paid, Unpaid.



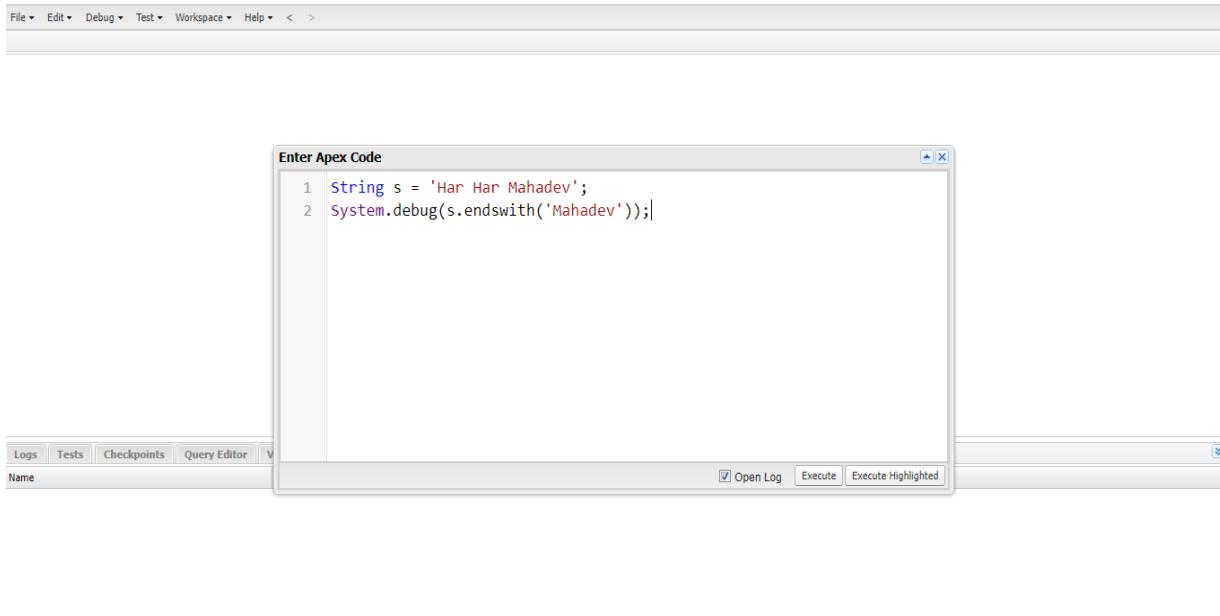
Test The App

- Code Playground App should look like this:

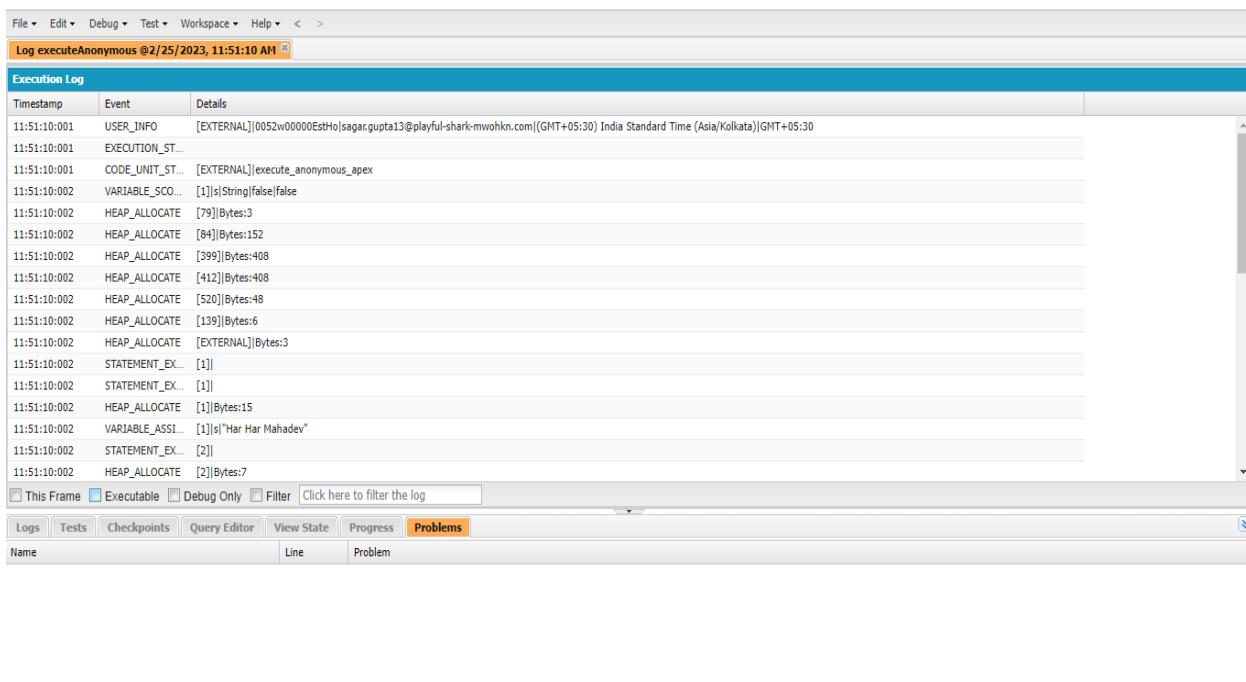
All Accounts							New	Import	Printable View
	Account Name	Account Site	Billing State/Province	Phone	Type	Account Owner Alias			
1	Burlington Textiles Corp of America		NC	(336) 222-7000	Customer - Direct	SGupt			
2	Dickenson plc		KS	(785) 241-6200	Customer - Channel	SGupt			
3	Edge Communications		TX	(512) 757-6000	Customer - Direct	SGupt			
4	Express Logistics and Transport		OR	(503) 421-7800	Customer - Channel	SGupt			
5	GenePoint		CA	(650) 867-3450	Customer - Channel	SGupt			
6	Grand Hotels & Resorts Ltd		IL	(312) 596-1000	Customer - Direct	SGupt			
7	Pyramid Construction Inc.			(014) 427-4427	Customer - Channel	SGupt			
8	Sample Account for Entitlements					autproc			
9	sForce		CA	(415) 901-7000		SGupt			
10	United Oil & Gas Corp.		NY	(212) 842-5500	Customer - Direct	SGupt			
11	United Oil & Gas, Singapore		Singapore	(650) 450-8810	Customer - Direct	SGupt			
12	United Oil & Gas, UK		UK	+44 191 4956203	Customer - Direct	SGupt			
13	University of Arizona		AZ	(520) 773-9050	Customer - Direct	SGupt			

Exercise 2 :- Use Execute Anonymous to define and execute the following code:

Code 1:- Define a String Variable & use string method ‘endsWith’ to display the output.



Output :



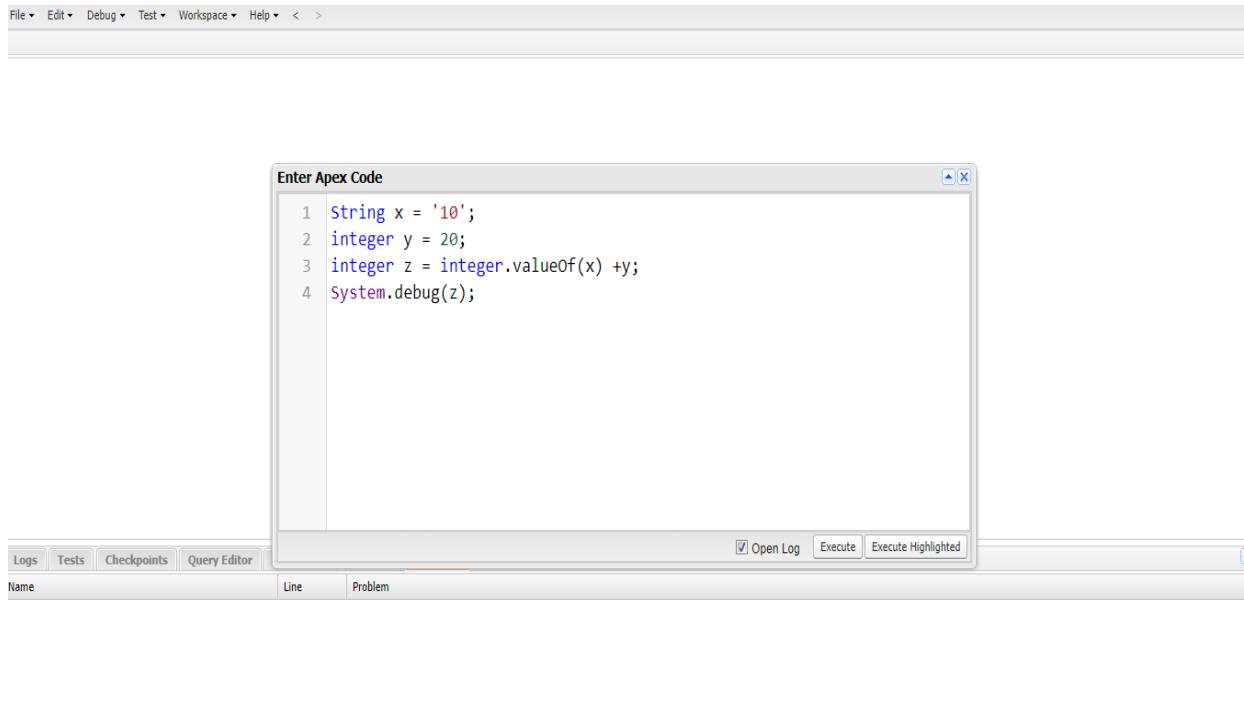
Code 2 : Define 2 Date type variables, use Date method today() & addDays(30) to display the output.

```
1 Date date1 = Date.today();
2 Date date2 = date1.adddays(30);
3 System.debug(date2);
```

Output :

Timestamp	Event	Details
12:00:59:003	USER_DEBUG	[3] DEBUG 2023-03-27 00:00:00

Code 4 : Display the output of an Integer variable from string '10' and then add 20 to it.

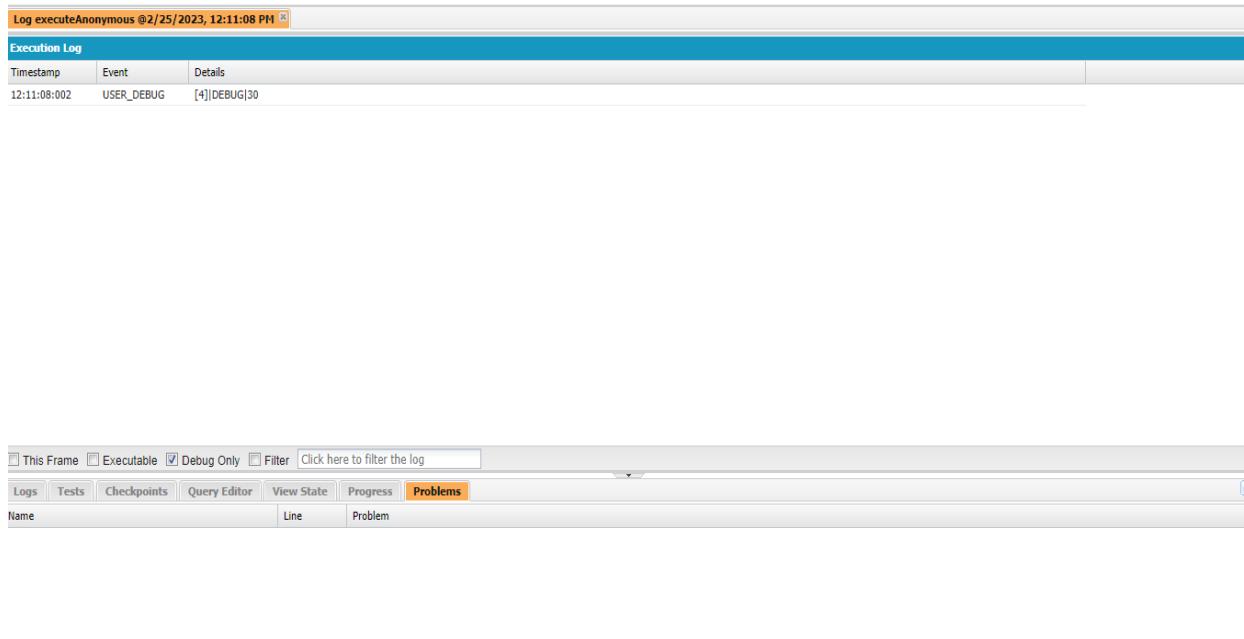


The screenshot shows the Salesforce IDE interface. At the top, there is a menu bar with options: File, Edit, Debug, Test, Workspace, Help, and several navigation icons. Below the menu is a toolbar with buttons for Logs, Tests, Checkpoints, and Query Editor. The main area contains a modal window titled "Enter Apex Code". Inside the modal, the following Apex code is displayed:

```
1 String x = '10';
2 integer y = 20;
3 integer z = integer.valueOf(x) +y;
4 System.debug(z);
```

Below the code editor are three buttons: "Open Log" (with a checked checkbox), "Execute", and "Execute Highlighted". At the bottom of the modal is a status bar with tabs for Name, Line, and Problem. The background of the IDE is mostly white, with some light gray shading around the modal window.

Output :



The screenshot shows the Salesforce IDE interface again. At the top, there is a header bar with a "Log executeAnonymous @2/25/2023, 12:11:08 PM" message. Below the header is a "Execution Log" tab. The log table has columns for Timestamp, Event, and Details. One entry is visible: "12:11:08:002 USER_DEBUG [4]DEBUG|30".

At the bottom of the interface, there is a "Logs" tab selected, along with other tabs: Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is highlighted with an orange border. A status bar at the very bottom shows tabs for Name, Line, and Problem.

Code 5 : Define a String Variable & use string method length() to display the output.

The screenshot shows the Salesforce IDE interface. At the top, there are three tabs: "Log executeAnonymous @2/25/2023, 12:11:08 PM" (disabled), "Log executeAnonymous @2/25/2023, 12:14:54 PM" (disabled), and "Log executeAnonymous @2/25/2023, 12:14:54 PM" (highlighted). Below the tabs is the "Execution Log" header with columns: "Timestamp", "Event", and "Details". A single log entry is shown: "12:14:54:002 USER_DEBUG [2]|DEBUG|15". The main area contains two windows: "Enter Apex Code" and "Logs". The "Enter Apex Code" window displays the following code:

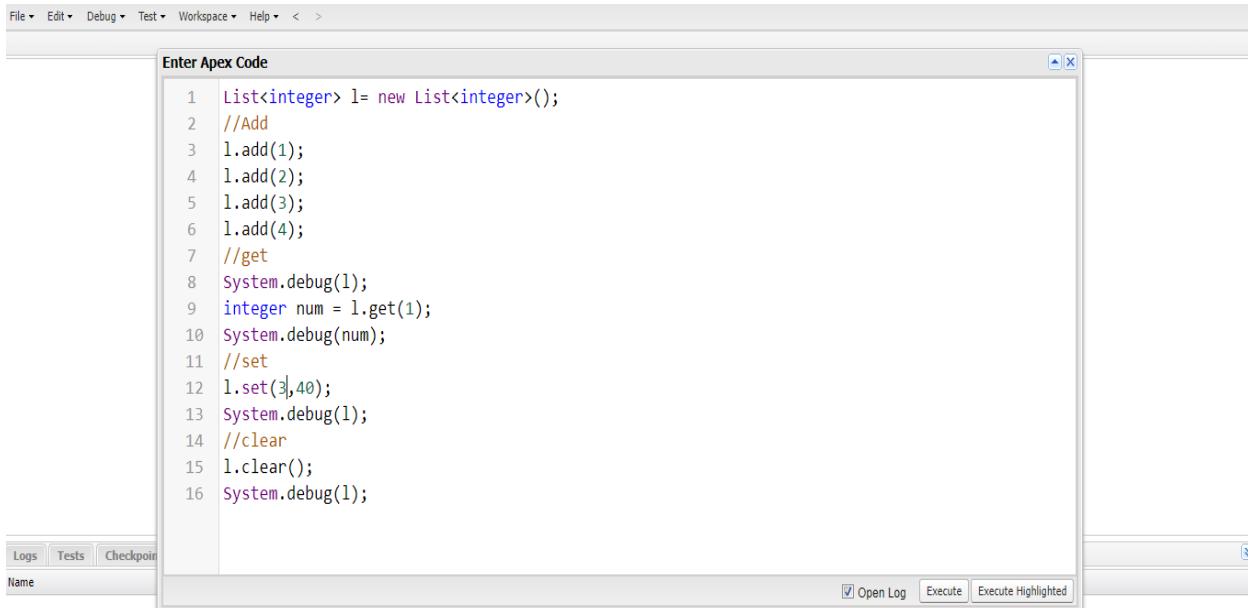
```
1 String x = 'Har har Mahadev';
2 System.debug(x.length());
```

The "Logs" window has tabs: "Logs", "Tests", "Checkpoints", "Query Editor", "View State", "Progress", and "Problems". The "Logs" tab is selected. It includes checkboxes for "This Frame", "Executable", and "Debug Only", and a "Filter" input field. Below the tabs is a table with columns: "Name", "Line", and "Problem".

Output :

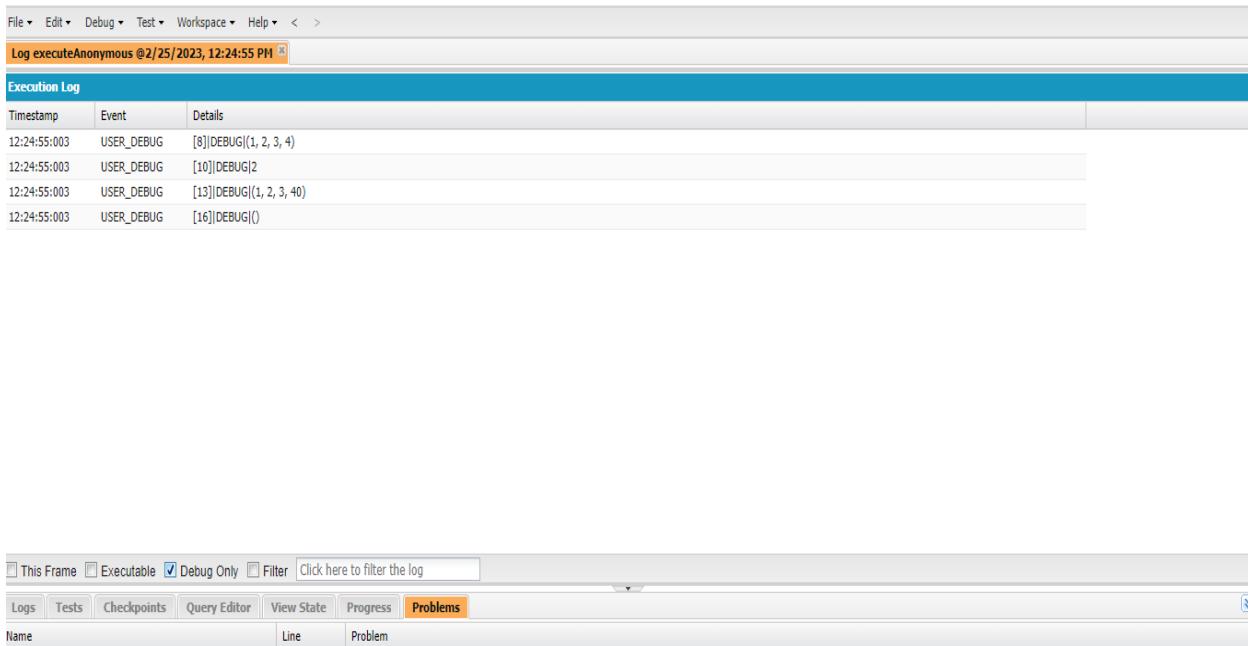
The screenshot shows the Salesforce IDE interface. At the top, there are three tabs: "Log executeAnonymous @2/25/2023, 12:11:08 PM" (disabled), "Log executeAnonymous @2/25/2023, 12:14:54 PM" (disabled), and "Log executeAnonymous @2/25/2023, 12:15:57 PM" (highlighted). Below the tabs is the "Execution Log" header with columns: "Timestamp", "Event", and "Details". A single log entry is shown: "12:15:57:003 USER_DEBUG [2]|DEBUG|15". The main area contains two windows: "Logs" and "Problems". The "Logs" window has tabs: "Logs", "Tests", "Checkpoints", "Query Editor", "View State", "Progress", and "Problems". The "Problems" tab is selected. It includes checkboxes for "This Frame", "Executable", and "Debug Only", and a "Filter" input field. Below the tabs is a table with columns: "Name", "Line", and "Problem".

Code 6 : Define a List of integer and display the output using add(), get(), set(), clear(), methods



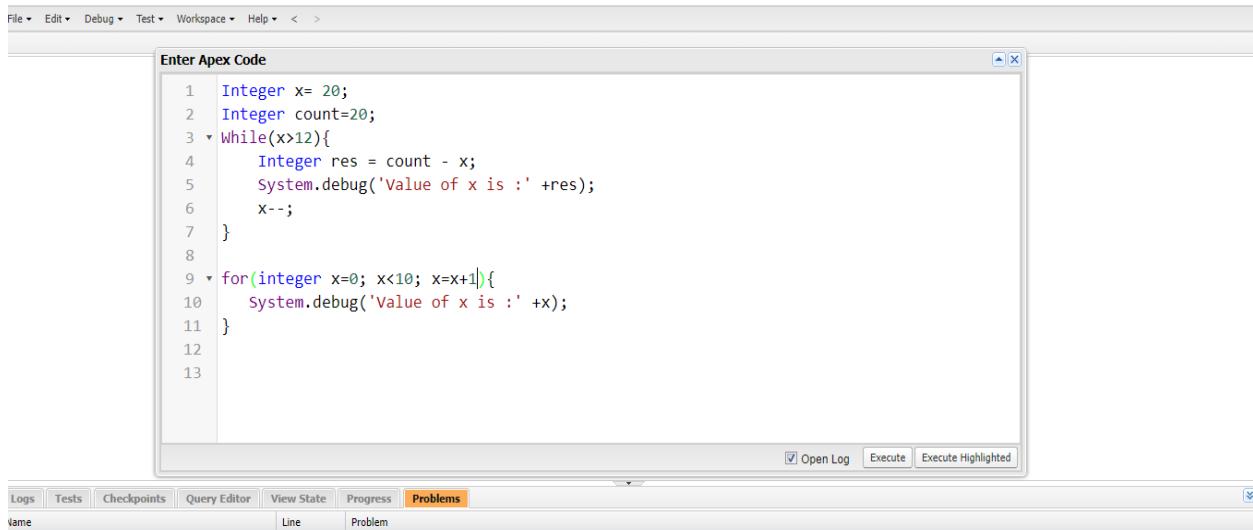
```
1 List<integer> l= new List<integer>();
2 //Add
3 l.add(1);
4 l.add(2);
5 l.add(3);
6 l.add(4);
7 //get
8 System.debug(l);
9 integer num = l.get(1);
10 System.debug(num);
11 //set
12 l.set(3,40);
13 System.debug(l);
14 //clear
15 l.clear();
16 System.debug(l);
```

Output :



Timestamp	Event	Details
12:24:55:003	USER_DEBUG	[8]DEBUG([1, 2, 3, 4])
12:24:55:003	USER_DEBUG	[10]DEBUG[2]
12:24:55:003	USER_DEBUG	[13]DEBUG([1, 2, 3, 40])
12:24:55:003	USER_DEBUG	[16]DEBUG()

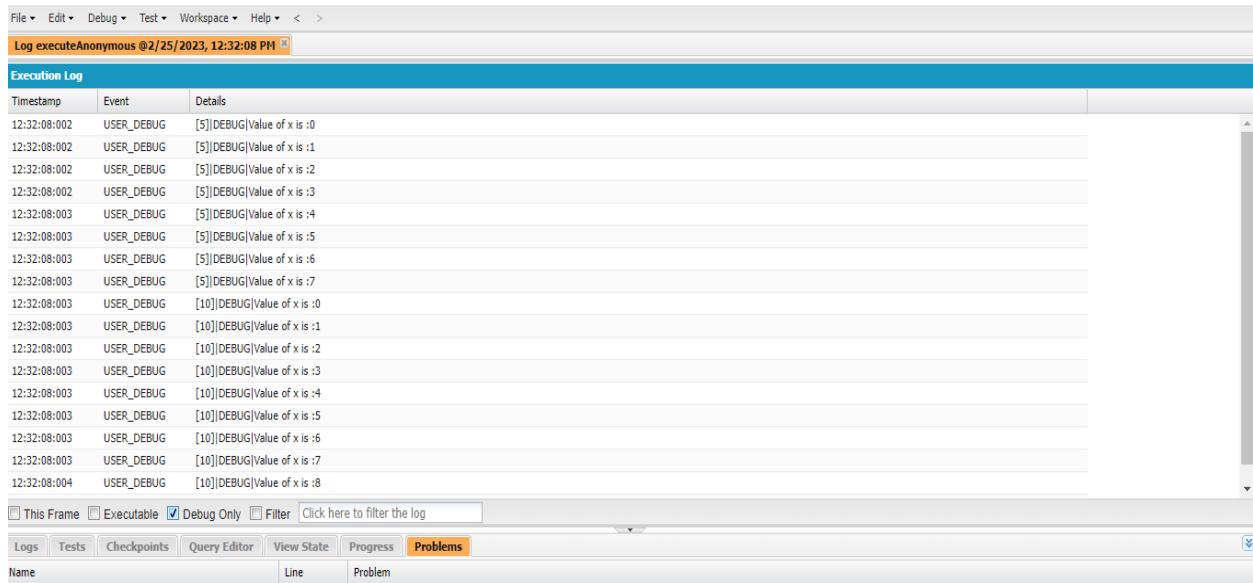
Code 7 : Use Execute Anonymous to define and execute the following code to display the value of x = 0 to 9.



```
1 Integer x= 20;
2 Integer count=20;
3 while(x>12){
4     Integer res = count - x;
5     System.debug('Value of x is :' +res);
6     x--;
7 }
8
9 for(integer x=0; x<10; x=x+1){
10     System.debug('Value of x is :' +x);
11 }
```

The screenshot shows the Salesforce IDE's Enter Apex Code window. The code is a combination of two snippets. The first snippet uses a while loop to decrement x from 20 down to 13, printing each value. The second snippet uses a for loop to increment x from 0 to 9, printing each value. Both snippets use System.debug to output the value of x.

Output :



Timestamp	Event	Details
12:32:08:002	USER_DEBUG	[5]DEBUG Value of x is :0
12:32:08:002	USER_DEBUG	[5]DEBUG Value of x is :1
12:32:08:002	USER_DEBUG	[5]DEBUG Value of x is :2
12:32:08:002	USER_DEBUG	[5]DEBUG Value of x is :3
12:32:08:003	USER_DEBUG	[5]DEBUG Value of x is :4
12:32:08:003	USER_DEBUG	[5]DEBUG Value of x is :5
12:32:08:003	USER_DEBUG	[5]DEBUG Value of x is :6
12:32:08:003	USER_DEBUG	[5]DEBUG Value of x is :7
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :0
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :1
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :2
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :3
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :4
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :5
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :6
12:32:08:003	USER_DEBUG	[10]DEBUG Value of x is :7
12:32:08:004	USER_DEBUG	[10]DEBUG Value of x is :8

The screenshot shows the Salesforce IDE's Execution Log window. It displays a log of debug statements from the executed code. The log shows the value of x being printed from 0 to 8, followed by values from 5 to 13 as the while loop continues.

Exercise 3 : - Answer the following in True Or False:

The screenshot shows the Salesforce IDE interface. At the top is a menu bar with File, Edit, Debug, Test, Workspace, Help, and navigation buttons. Below the menu is a toolbar with icons for Save, Undo, Redo, Cut, Copy, Paste, Find, Replace, and Delete. The main area is titled "Enter Apex Code" and contains the following Apex code:

```
1 Integer myunluckyNumber = 7;
2 Integer myluckyNumber = 15;
3 System.debug(myluckyNumber != myunluckyNumber + 8);
```

At the bottom of the code editor are three buttons: Open Log, Execute, and Execute Highlighted. Below the code editor is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is highlighted. A sub-menu below the navigation bar shows "Name" and "Line" options.

Output :

The screenshot shows the Salesforce IDE interface with the Execution Log panel open. The title bar says "Log executeAnonymous @2/25/2023, 12:35:10 PM". The Execution Log table has columns for Timestamp, Event, and Details. One entry is shown:

Timestamp	Event	Details
12:35:10:002	USER_DEBUG	[3]DEBUG:false

Below the table are filter options: This Frame, Executable, Debug Only, Filter, and Click here to filter the log. At the bottom is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is highlighted. A sub-menu below the navigation bar shows "Name" and "Line" options.

Exercise 4 : - Answer the following in True Or False:

The screenshot shows the Salesforce IDE interface. At the top is a menu bar with File, Edit, Debug, Test, Workspace, Help, and navigation icons. Below the menu is a toolbar with standard file operations. The main area is titled "Enter Apex Code" and contains the following Apex code:

```
1 Boolean.isTrue = True;
2 Boolean.isFalse = false;
3 System.debug(isTrue || isFalse);
```

At the bottom of the code editor are three buttons: Open Log, Execute, and Execute Highlighted. Below the code editor is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected and highlighted in orange. Underneath the navigation bar is a search bar labeled "Name" and filter buttons for "Line" and "Problem".

Output :

The screenshot shows the Salesforce IDE interface with the "Execution Log" panel open. The title bar indicates "Log executeAnonymous @2/25/2023, 12:38:19 PM". The execution log table has columns for Timestamp, Event, and Details. One entry is visible:

Timestamp	Event	Details
12:38:19:002	USER_DEBUG	[3] DEBUG true

Below the execution log is a "Log" panel with a header containing checkboxes for "This Frame", "Executable", "Debug Only", and "Filter", followed by a "Click here to filter the log" link. The log panel also features a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected and highlighted in orange. A search bar labeled "Name" and filter buttons for "Line" and "Problem" are located at the bottom of the log panel.

Exercise 5 : - Answer the following in True Or False:

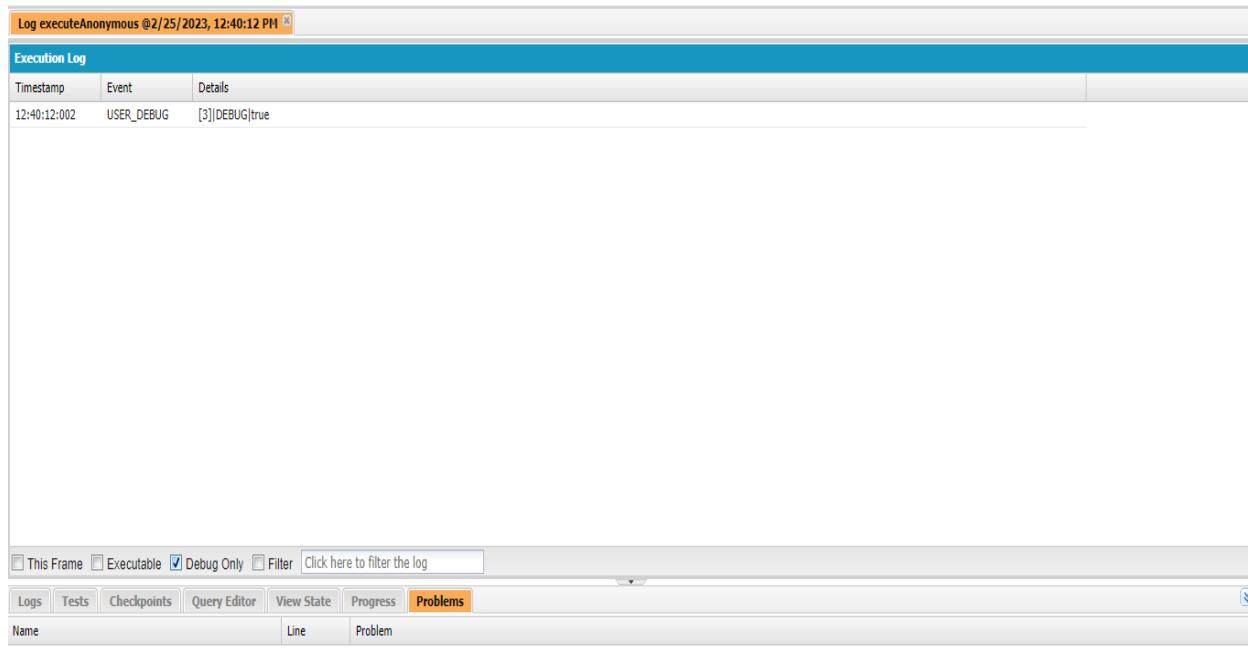


The screenshot shows the Salesforce IDE interface. At the top is a menu bar with File, Edit, Debug, Test, Workspace, Help, and navigation icons. Below the menu is a toolbar with standard file operations. The main area contains a window titled "Enter Apex Code" with the following Apex code:

```
1 Date today = Date.today();
2 Date tomorrow = Date.today().addDays(1);
3 System.debug(today != tomorrow);
```

At the bottom of the code editor are three buttons: Open Log, Execute, and Execute Highlighted. Below the code editor is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected and highlighted in orange. Underneath the navigation bar is a search bar labeled "Name".

Output :



The screenshot shows the Execution Log window. The title bar says "Log executeAnonymous @2/25/2023, 12:40:12 PM". The main area is titled "Execution Log" and contains a table with three columns: Timestamp, Event, and Details. One entry is visible:

Timestamp	Event	Details
12:40:12:002	USER_DEBUG	[3]DEBUG true

Below the log table are several filter options: "This Frame", "Executable", "Debug Only" (which is checked), "Filter", and "Click here to filter the log". At the bottom is another navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected and highlighted in orange. Underneath the navigation bar is a search bar labeled "Name".

Exercise 6 : - Write a program and execute to demo the use of “If..else if...else”.

The screenshot shows the Salesforce Apex code editor. The main window displays the following Apex code:

```
1 Integer Score =80;
2 If (Score == 100){
3     System.debug('Grade: A+');
4 }else If (Score >= 90){
5     System.debug('Grade: A');
6 }else If (Score >= 80){
7     System.debug('Grade: B');
8 }else{
9     System.debug('Grade:Failed');
10 }
```

Below the code editor, there are three buttons: "Open Log", "Execute", and "Execute Highlighted".

At the bottom of the interface, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is currently selected. Below the navigation bar, there is a search bar with fields for "Name", "Line", and "Problem".

Output :

The screenshot shows the Salesforce Execution Log. The log title is "Log executeAnonymous @2/25/2023, 12:45:28 PM". The log table has columns: Timestamp, Event, and Details. One entry is visible:

Timestamp	Event	Details
12:45:28:002	USER_DEBUG	[7]DEBUG Grade: B

At the top of the log area, there are checkboxes for "This Frame", "Executable", "Debug Only", and "Filter", along with a link "Click here to filter the log".

Below the log, there is a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is currently selected. There is also a search bar with fields for "Name", "Line", and "Problem".

Exercise 7 : - Write a program to execute and demo the use of “Apex – for Loop”

Note: Create at least 2 Billing records with Status = Paid for this exercise.

The screenshot shows the Salesforce Code Playground interface. At the top, there's a navigation bar with links like Accounts, Leads, Contacts, Opportunities, Cases, Customers, and Billings. Below the navigation is a search bar and a toolbar with icons for New, Import, Change Owner, and Printable View. The main area displays a list of Billings with columns for Bill Number, Customer Type, and Status. There are four items listed: B-0001 (Premium, Paid), B-0002 (Standard, Paid), B-0003 (Premium, Unpaid), and B-0004 (Standard, Unpaid). Below this is the code editor for Billing.apxc:

```
1 public class Billing {
2     public static void viewbills(){
3         List<Billing__c> BillingList=[SELECT Id, Name, Status__c FROM Billing__c];
4         List<String> Billsrecord = new List<String>();
5         for(Billing__c bill : BillingList){
6             System.debug ('Value of Current Record in the Loop' + BillingList);
7             if(bill.Status__c == 'Paid'){
8                 Billsrecord.add(bill.name);
9             }
10        }
11        System.debug('Value of BillingList '+Billsrecord);
12    }
13 }
14 }
```

At the bottom of the code editor, there are tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected. The output pane below shows the execution log:

Timestamp	Event	Details
13:10:44:067	USER_DEBUG	[6]DEBUG Value of Current Record in the Loop(Billing__c:{Id=a012w000017Z6EXAA0, Name=B - 0002, Status__c=Paid}, Billing__c:{Id=a012w000017Z6ESAA0, Name=B - 0001, Status__c=Paid})
13:10:44:068	USER_DEBUG	[6]DEBUG Value of Current Record in the Loop(Billing__c:{Id=a012w000017Z6EXAA0, Name=B - 0002, Status__c=Paid}, Billing__c:{Id=a012w000017Z6ESAA0, Name=B - 0001, Status__c=Paid})
13:10:44:068	USER_DEBUG	[11]DEBUG Value of BillingList (B - 0002, B - 0001)

Output :

The screenshot shows the Salesforce Debug Console. At the top, there's a menu bar with File, Edit, Debug, Test, Workspace, Help, and a log header for Billing.apxc Log executeAnonymous @2/25/2023, 1:10:44 PM. Below the header is an Execution Log table with columns for Timestamp, Event, and Details. The log entries are identical to those in the Code Playground. Below the log is an Enter Apex Code editor window containing the code: billing.viewbills();. At the bottom of the screen, there's a toolbar with checkboxes for This Frame, Executable, Debug Only, Filter, and Click here to filter the log, along with buttons for Open Log, Execute, and Execute Highlighted. The bottom also features tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems, with the Problems tab selected.

Exercise 8 : - Write a Class to demo the use of Constants in Apex

The screenshot shows the Salesforce IDE interface. The top menu bar includes File, Edit, Debug, Test, Workspace, Help, and a search bar. Below the menu is a toolbar with Code Coverage: None, API Version: 57, and a Go To button. The main area displays the code for `DiscountClass.apxc`:

```
1 public class DiscountClass {  
2     public static Decimal calculateDiscount(Integer price){  
3         Decimal regularDiscount =0.1;  
4         Decimal finalPrice = price - price*regularDiscount;  
5         return finalPrice;  
6     }  
7 }|
```

At the bottom of the code editor, there are tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected, showing a table with columns Name, Line, and Problem.

Output :

The screenshot shows the Salesforce IDE interface with the Execution Log and Enter Apex Code dialog open.

Execution Log:

Timestamp	Event	Details
13:21:51:011	USER_DEBUG	[2]DEBUG finalPrice90.0

Enter Apex Code:

```
1 Decimal finalPrice= DiscountClass.calculateDiscount(100);  
2 System.debug('finalPrice' +finalPrice);|
```

Below the dialogs, there is a log viewer with checkboxes for This Frame, Executable, Debug Only, Filter, and a Click here to filter the log button. It also has tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems, with the Problems tab selected.

Exercise 9 : - Write a Class to demo the use of Interface in Apex

The image shows three vertically stacked code editors from the Salesforce Trailhead interface, each displaying a different Apex class:

- InterfaceExample.apxc:** Contains the definition of a public interface named InterfaceExample with a single method signature: Double percentageDiscountTobeApplied();.
- PremiumCustomer.apxc:** Contains the implementation of the InterfaceExample interface for Premium Customer. It defines a public class PremiumCustomer implements InterfaceExample with a method returning 0.30.
- normalCustomer.apxc:** Contains the implementation of the InterfaceExample interface for Normal Customer. It defines a public class normalCustomer implements InterfaceExample with a method returning 0.10.

Each code editor includes standard Salesforce development tools like tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems, and a Go To button.

Output :

The screenshot shows the Salesforce Apex CS1 Page interface. At the top, there are tabs for InterfaceExample.apxc, PremiumCustomer.apxc, and normalCustomer.apxc. Below the tabs, the title bar includes File, Edit, Debug, Test, Workspace, Help, and a search bar. The main area has a header "Execution Log" with columns for Timestamp, Event, and Details. Two log entries are shown:

Timestamp	Event	Details
16:37:56:026	USER_DEBUG	[3]DEBUG Discount in Percentage From premium30.0%
16:37:56:027	USER_DEBUG	[5]DEBUG Discount in Percentage From Normal10.0%

Below the log is a modal window titled "Enter Apex Code" containing the following Apex code:

```
1 PremiumCustomer p1=new PremiumCustomer();
2 Double discount = p1.percentageDiscountToBeApplied();
3 System.debug('Discount in Percentage From premium'+(discount*100)+'%');
4 normalCustomer n1=new normalCustomer();
5 discount = n1.percentageDiscountToBeApplied();
6 System.debug('Discount in Percentage From Normal'+(discount*100)+'%');
```

At the bottom of the modal are buttons for Open Log, Execute, and Execute Highlighted. The footer of the page includes links for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems.

Exercise 10 : - Demo on DML Insert Operation Using Database methods

The screenshot shows the Salesforce Apex CS1 Page interface. The title bar shows the file name DML.apxc and the log message Log executeAnonymous @2/26/2023, 12:08:53 AM. The code coverage is set to None and the API version is 57. The code editor contains the following Apex class:

```
1 public class DML {
2     Public void test()
3     {
4         Customer__c cust = new Customer__c();
5         cust.name = 'Wipro';
6         cust.Customer_Type__c = 'Premium';
7
8         insert cust;
9
10        List<Billing__c> listinsert = new List<Billing__c>{new Billing__c(Status__c = 'paid',Amount_Paid__c = 5000000)};
11        Database.SaveResult[] srList = Database.insert(listinsert,false);
12        For(Database.SaveResult sr : srList){
13            if(sr.isSuccess()){
14                System.debug('Successfully inserted Billing'+sr.getId());
15            }else{
16                for(Database.Error err : sr.getErrors()){
17                    System.debug('the Following error has Occurred.');
18                    System.debug(err.getStatuscode() + ':' +err.getMessage());
19                    System.debug('Billing object Field which are Affect by the error :'+err.getFields());
20
21                }
22            }
23        }
24    }
25 }
```

The code uses the Database class to insert a new Customer record and a list of Billing records. It then iterates through the save results to debug success or failure messages and specific error details.

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

DHLLapic [Log executeAnonymous @2/25/2023, 11:37:30 PM]

Execution Log

Timestamp	Event	Details
23:37:30:001	USER_INFO	[EXTERNAL][0052w00000Eth0 sagar.gupta13@playful-shark-mwohkn.com](GMT+05:30) India Standard Time (Asia/Kolkata)[GMT+05:30]
23:37:30:002	EXECUTION_ST...	
23:37:30:002	CODE_UNIT_ST...	[EXTERNAL]execute_anonymous_apex
23:37:30:002	VARIABLE_SO...	[1]cust Customer__c true false
23:37:30:002	HEAP_ALLOCATE	[79]Bytes:3
23:37:30:002	HEAP_ALLOCATE	[84]Bytes:152
23:37:30:002	HEAP_ALLOCATE	[399]Bytes:408
23:37:30:002	HEAP_ALLOCATE	[412]Bytes:408
23:37:30:002	HEAP_ALLOCATE	[520]Bytes:48
23:37:30:002	HEAP_ALLOCATE	[139]Bytes:6
23:37:30:002	HEAP_ALLOCATE	[EXTERNAL]Bytes:7
23:37:30:003	STATEMENT_EX...	[1]
23:37:30:003	STATEMENT_EX...	[1]
23:37:30:003	HEAP_ALLOCATE	[1]Bytes:4
23:37:30:003	VARIABLE_ASSI...	[1]cust:{} 0x7d6306ae
23:37:30:003	STATEMENT_EX...	[2]
23:37:30:003	HEAP_ALLOCATE	[2]Bytes:5
23:37:30:003	VARIABLE_ASSI...	[2]this.Name Wipro 0x7d6306ae
23:37:30:003	STATEMENT_EX...	[3]
23:37:30:003	HEAP_ALLOCATE	[3]Bytes:7
23:37:30:004	VARIABLE_ASSI...	[3]this.Customer_Type__c Premium 0x7d6306ae
23:37:30:004	STATEMENT_EX...	[4]
23:37:30:004	HEAP_ALLOCATE	[52]Bytes:5

This Frame Executable Debug Only Filter Click here to filter the log

Enter Apex Code

```

10 Customer__c cust = new Customer__c();
11 cust.name = 'Wipro';
12 cust.Customer_Type__c = 'Premium';
13 insert cust;
14
15
16
17
18
19
20
21
22

```

Open Log Execute Execute Highlighted

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

Output :

← → C playful-shark-mwohkn-dev-ed.trailblaze.lightning.force.com/lightning/o/Customer__c/list?filterName=00B2w00000Ydu0kEAB

<code>

Code Playground Accounts Leads Contacts Opportunities Cases Customers Billings

Customers All

1 Item • Sorted by Customer Name • Filtered by All customers • Updated a few seconds ago

Customer Name	Customer Type
Wipro	Premium

New Import Change Owner Printable View

Search this list...

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

DHLLapic [Log executeAnonymous @2/26/2023, 12:08:53 AM]

Execution Log

Timestamp	Event	Details
00:08:53:041	USER_DEBUG	[\$]DEBUGSuccessfully inserted Billing@012w00000172beAA0

Enter Apex Code

```

1 List<Billing__c> listinsert = new List<Billing__c>{new Billing__c(Status__c = 'paid',Amount_Paid__c = 5000000)};
2 Database.SaveResult[] srlist = Database.insert(listinsert, false);
3 For(Database.SaveResult sr : srlist){
4     if(sr.isSuccess()){
5         System.debug('Successfully inserted Billing'+sr.getId());
6     }else{
7         for(Database.Error err : sr.getErrors()){
8             System.debug('the Following error has Occurred.');
9             System.debug(err.getStatusCode()+' : '+err.getMessage());
10            System.debug('Billing object Field which are Affect by the error :'+err.getFields());
11        }
12    }
13 }
14
15
16

```

Open Log Execute Execute Highlighted

This Frame Executable Debug Only Filter Click here to filter the log

Logs Tests Checkpoints Query Editor View State Progress Problems

Name Line Problem

Output :

The screenshot shows the Salesforce Billings detail page for record B - 0005. The page includes fields for Bill Number, Amount Paid, Customer Type, Status, and Created By. It also displays the Owner (Sagar Gupta) and Last Modified By (Sagar Gupta). On the right, there is an Activity sidebar showing no upcoming or overdue activities.

Exercise 11 :- Write and execute SOQL queries from Developer Console.

Output :

The screenshot shows the Salesforce Developer Console. A SOQL query is run to select opportunities where Account.Industry = 'Energy' and Account.AnnualRevenue > 5000. The results table shows 10 rows of data. Below the query editor, the history pane shows the executed query.

ID	Amount	StageName	Account.Name	Account.Industry	Account.Website
0062w00000KCU1tAAH	125000	Negotiation/Review	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU1yAAH	270000	Proposal/Price Quote	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU1yAAH	120000	Closed Won	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU26AAH	270000	Negotiation/Review	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU29AAH	270000	Closed Won	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU2BAAX	915000	Closed Won	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU2GAAX	235000	Closed Won	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU2HAAX	440000	Closed Won	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU2JAAX	120000	Closed Won	United Oil & Gas Corp.	Energy	http://www.uos.com
0062w00000KCU2LAAX	675000	Needs Analysis	United Oil & Gas Corp.	Energy	http://www.uos.com

Exercise 12 :- Write an Apex Trigger, Name = CustomerTrigger.

The screenshot shows the Salesforce IDE interface with the code editor open. The file is named CustomerTrigger.apxt. The code implements a trigger on the Customer__c object that inserts a new Billing__c record for each customer if their Active__c field is false. The Billing__c record has a status of 'Paid' and an amount of 1000000. The trigger also inserts the Billinglist.

```
trigger CustomerTrigger on Customer__c (after insert, after update) {
    List<Billing__c> BillingList = new List<Billing__c>();
    for (Customer__c objCustomer: Trigger.new)
    {
        if (objCustomer.Active__c == False)
        {
            Billing__c objbill = new Billing__c();
            objbill.Status__c = 'Paid';
            objbill.Amount_Paid__c=1000000;
            BillingList.add(objbill);
        }
    }
    insert BillingList;
}
```

Exercise 13 :- Write a Test Class for Customer Trigger.

Output :

The screenshot shows the Salesforce IDE interface with the CustomerTriggerTest.apxc test class open. The class contains a single test method named testName that creates a new Customer__c record with Active__c set to false, updates it to true, and then runs a test. The test passes. Below the code editor is a Test Results table showing the run details.

```
@isTest
public class CustomerTriggerTestClass {
    @isTest static void testName() {
        Customer__c cust = new Customer__c();
        cust.Active__c = False;
        insert cust;

        Test.startTest();
        cust.Active__c = True;
        update cust;
        Test.stopTest();
    }
}
```

Status	Test Run	Enqueued Time	Duration	Failures	Total
✖	7072w00008M89d	Sat Feb 25 2023 22:59:53 GMT...	0:00	1	2
✓	7072w00008M89v	Sat Feb 25 2023 22:04:49 GMT...	0:00	0	2
✖	7072w00008M8Gq	Sat Feb 25 2023 22:57:21 GMT...	0:00	1	3
✓	7072w00008M8Gg	Sat Feb 25 2023 22:58:00 pm	0:00	0	1
✓	7072w00008M8Gg	DisqualifyTestLeads	0:00	0	1
✓	a		0:00	0	1
✖	7072w00008M8MK	Sat Feb 25 2023 22:59:25 GMT...	0:00	1	4

Overall Code Coverage

Class	Percent	Lines
Overall	35%	
Billing	0%	0/6
CustomerTrigger	100%	9/9
demo	0%	0/4
DiscountClass	0%	0/4

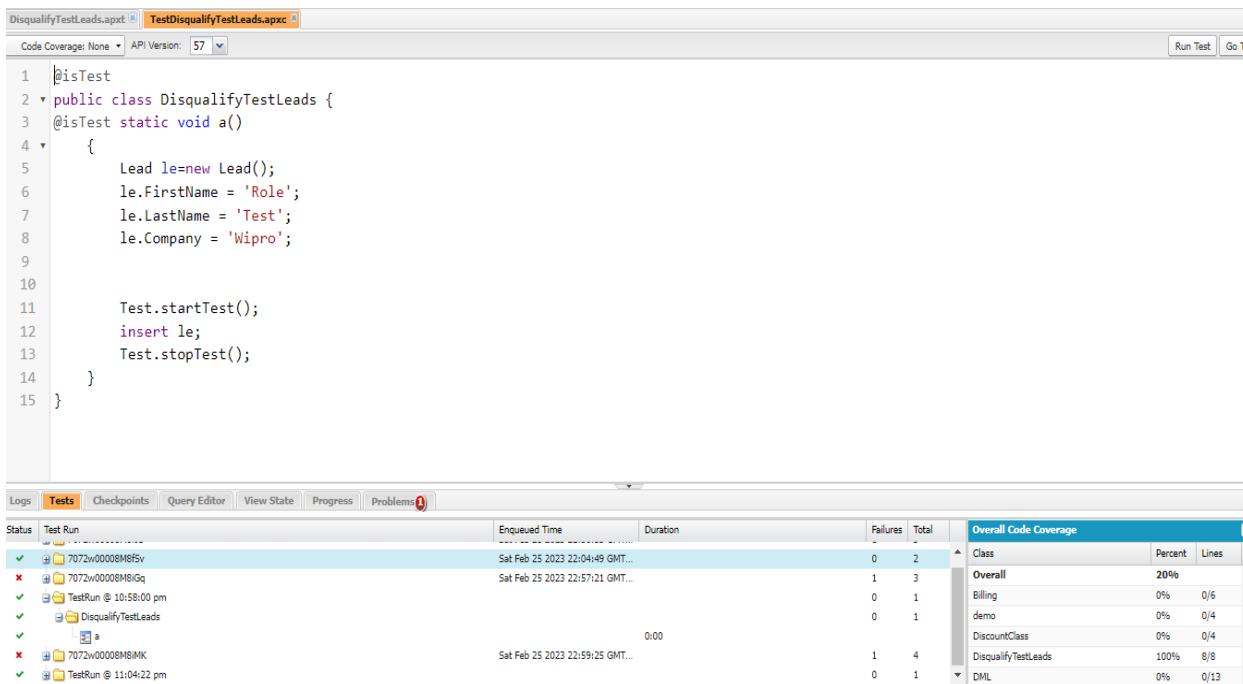
Exercise 14 :- Write an Apex Trigger, Name = DisqualifyTestLeads.



```
DisqualifyTestLeads.apxt TestDisqualifyTestLeads.apxc
Code Coverage: None API Version: 57 Go To
1 trigger DisqualifyTestLeads on Lead (before insert) {
2     List<Lead> llist = new List<Lead>();
3     for(Lead le:Trigger.new)
4     {
5         if(le.FirstName.containsIgnoreCase('test')|| string.isBlank(le.FirstName)
6             || le.LastName.containsIgnoreCase('test')||string.isBlank(le.LastName))
7         {
8             system.debug(le.FirstName + ' ' + le.LastName + 'Will be disqualified!');
9             llist.add(le);
10        }
11    }
12    for(Lead l :llist){
13        l.status='Disqualified';
14    }
15 }
```

Exercise 15 :- Write a Test Class for DisqualifyTestLeads.

Output :



```
DisqualifyTestLeads.apxt TestDisqualifyTestLeads.apxc
Code Coverage: None API Version: 57 Run Test Go To
1 @isTest
2 public class DisqualifyTestLeads {
3     @isTest static void a()
4     {
5         Lead le=new Lead();
6         le.FirstName = 'Role';
7         le.LastName = 'Test';
8         le.Company = 'Wipro';
9
10
11        Test.startTest();
12        insert le;
13        Test.stopTest();
14    }
15 }
```

Status	Test Run	Enqueued Time	Duration	Failures	Total	Overall Code Coverage
✓	7072w00008M85v	Sat Feb 25 2023 22:04:49 GMT...		0	2	Overall 20%
✗	7072w00008M8iQq	Sat Feb 25 2023 22:57:21 GMT...		1	3	Billing 0% 0/6
✓	TestRun @ 10:58:00 pm			0	1	demo 0% 0/4
✓	DisqualifyTestLeads			0	1	DiscountClass 0% 0/4
✓	__ a			1	4	DisqualifyTestLeads 100% 8/8
✗	7072w00008M8MK	Sat Feb 25 2023 22:59:25 GMT...	0:00	1	4	DML 0% 0/13
✓	TestRun @ 11:04:22 pm			0	1	

Exercise 16 :- Create a Visualforce page which displays Opportunity fields as output fields.

The screenshot shows the Salesforce IDE interface. At the top, there's a title bar with 'Opportunity.vfp' and a preview icon. Below it, a toolbar has 'Preview' and 'API Version: 57'. The main area contains the Visualforce page code:

```
1 <apex:page standardController = "Opportunity">
2
3     <apex:pageBlock title = "Opportunities">
4         <apex:pageBlockSection >
5             <apex:outputField value="{! Opportunity.Name}"/>
6             <apex:outputField value="{! Opportunity.Amount}"/>
7             <apex:outputField value="{! Opportunity.CloseDate}"/>
8             <apex:outputField value="{! Opportunity.Account.Name}"/>
9
10        </apex:pageBlockSection>
11    </apex:pageBlock>
12 </apex:page>
```

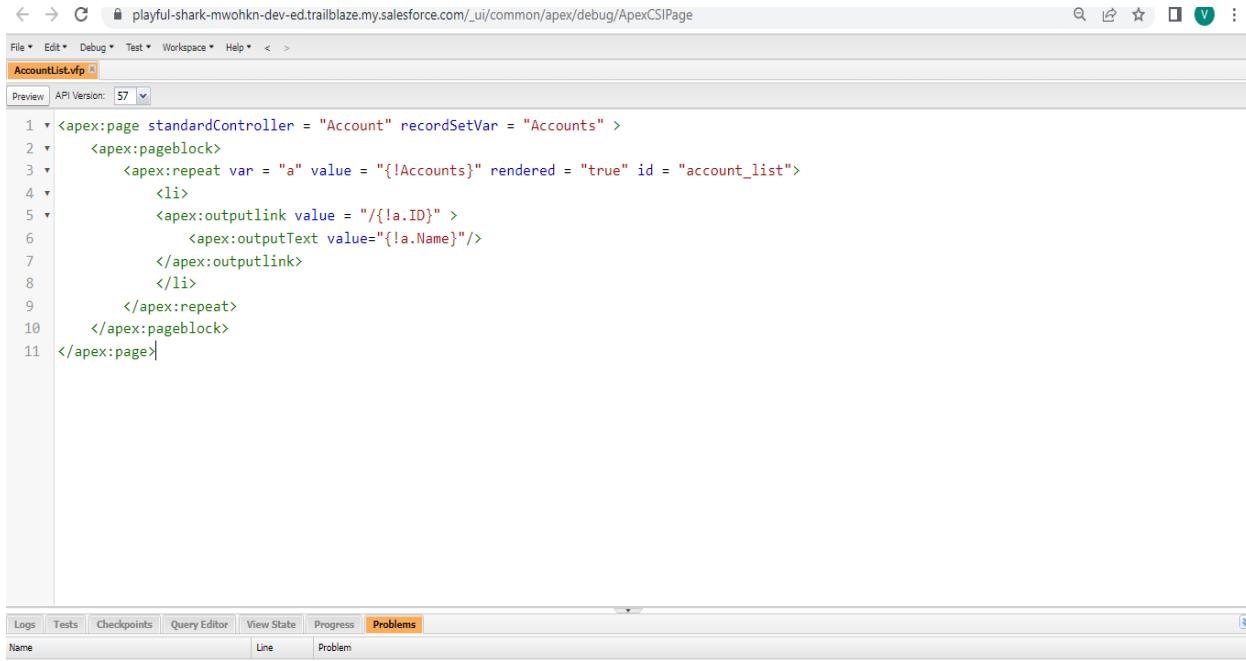
At the bottom, there's a navigation bar with tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The 'Problems' tab is selected, showing a table with columns: Name, Line, and Problem. There are no problems listed.

Output :

The screenshot shows a browser window with the URL playful-shark-mwohln-dev-ed--c.trailblaze.vf.force.com/apex/Opportunity?core.apexpages.request.devconsole=1. The page title is 'Opportunities'. The content area displays a table with four columns: Opportunity Name, Amount, Close Date, and Account Name. The table has two rows of placeholder data.

Opportunity Name	Amount
Close Date	Account Name
Opportunity 1	100000
Opportunity 2	200000

Exercise 17 :- Create a Visualforce page which shows a list of Accounts linked to their record page.

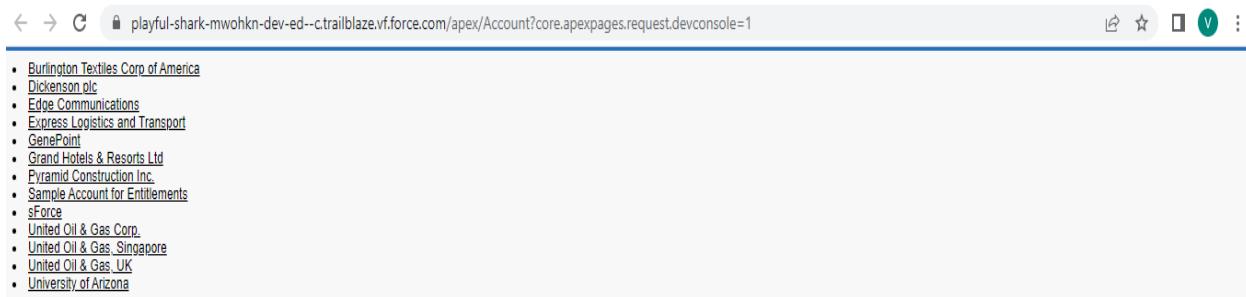


The screenshot shows the Salesforce Apex code editor interface. The title bar indicates the URL is playful-shark-mwohkn-dev-ed.trailblaze.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The tab bar shows 'AccountList.vfp'. The code area contains the following Apex code:

```
1 <apex:page standardController = "Account" recordSetVar = "Accounts" >
2   <apex:pageblock>
3     <apex:repeat var = "a" value = "{!Accounts}" rendered = "true" id = "account_list">
4       <li>
5         <apex:outputlink value = "/{!a.ID}" >
6           <apex:outputText value="{!!a.Name}" />
7         </apex:outputlink>
8       </li>
9     </apex:repeat>
10    </apex:pageblock>
11 </apex:page>
```

The bottom navigation bar includes tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Problems tab is selected, showing no errors or warnings.

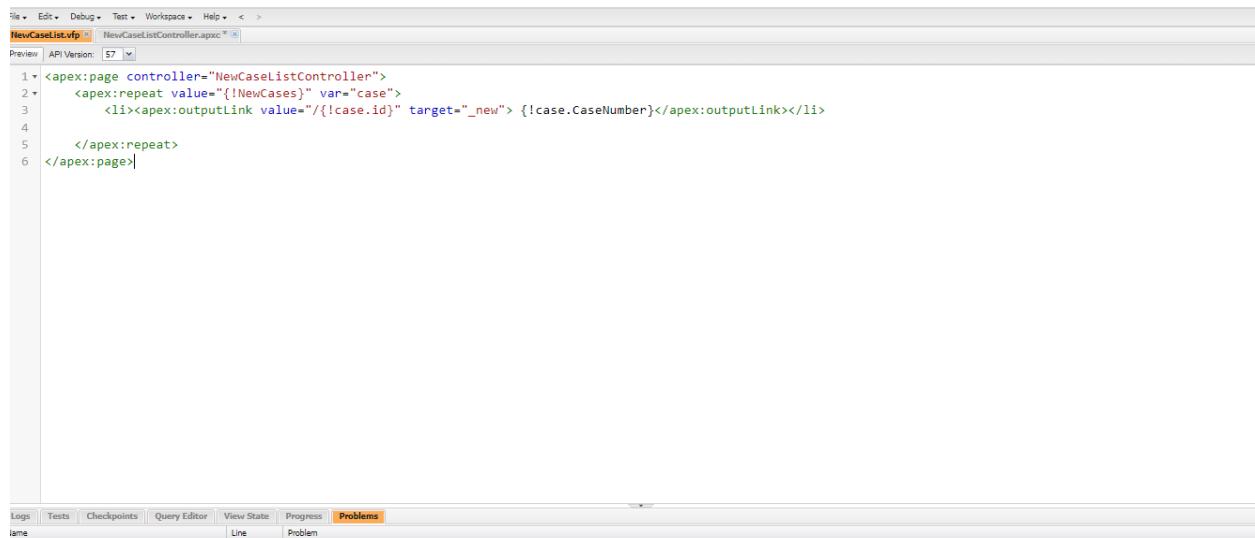
Output :



The screenshot shows the browser output for the AccountList.vfp page. The page displays a list of account names as hyperlinks:

- [Burlington Textiles Corp of America](#)
- [Dickenson plc](#)
- [Edge Communications](#)
- [Express Logistics and Transport](#)
- [GenePoint](#)
- [Grand Hotels & Resorts Ltd](#)
- [Pyramid Construction Inc.](#)
- [Sample Account for Entitlements](#)
- [sForce](#)
- [United Oil & Gas Corp.](#)
- [United Oil & Gas_Singapore](#)
- [United Oil & Gas_UK](#)
- [University of Arizona](#)

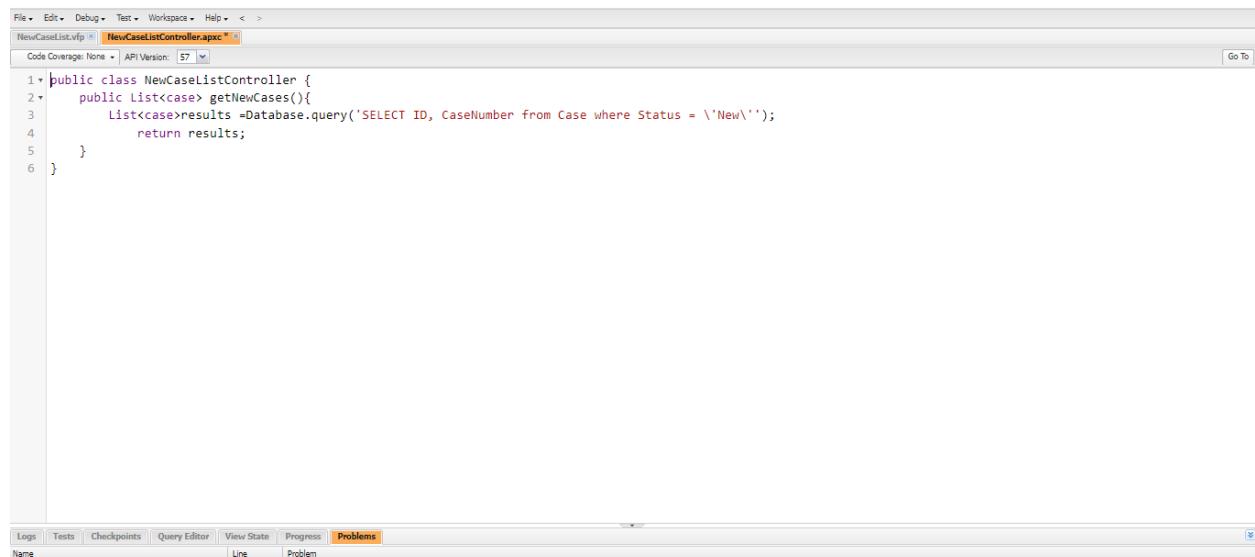
Exercise 18 :- Create a Visualforce page that uses a custom controller to display a list of cases with the status of 'New'. The page must be named NewCaseList.



The screenshot shows the Visualforce Editor interface. The top bar includes 'File', 'Edit', 'Debug', 'Test', 'Workspace', 'Help', and tabs for 'Preview' and 'API Version: 57'. The main area displays the Visualforce page code:

```
1 <apex:page controller="NewCaseListController">
2   <apex:repeat value="={!NewCases}" var="case">
3     <li><apex:outputLink value="/{!case.id}" target="_new"> {!case.CaseNumber}</apex:outputLink></li>
4   </apex:repeat>
5 </apex:page>
```

Below the code editor is a navigation bar with tabs: 'Logs', 'Tests', 'Checkpoints', 'Query Editor', 'View State', 'Progress', and 'Problems'. The 'Problems' tab is selected, showing no errors.

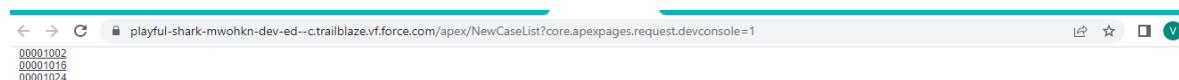


The screenshot shows the Apex Editor interface. The top bar includes 'File', 'Edit', 'Debug', 'Test', 'Workspace', 'Help', and tabs for 'Code Coverage: None' and 'API Version: 57'. The main area displays the Apex controller code:

```
1 public class NewCaseListController {
2   public List<Case> getNewCases(){
3     List<Case> results = Database.query('SELECT ID, CaseNumber FROM Case WHERE Status = \'New\'');
4     return results;
5   }
6 }
```

Below the code editor is a navigation bar with tabs: 'Logs', 'Tests', 'Checkpoints', 'Query Editor', 'View State', 'Progress', and 'Problems'. The 'Problems' tab is selected, showing no errors.

Output :



References

- 1.Trailhead: <https://trailhead.salesforce.com/>
2. Developer Guide: Salesforce Developers
- 3.[Manage sales - Salesforce IN](#)
- 4.[Salesforce - ADX201 Administrative Essentials for New Admins in Lightning Experience \(SFADX201\) \(qa.com\)](#)
- 5.[Understand the Salesforce Architecture Unit | Salesforce Trailhead](#)