CRM APPLICATION MANAGEMENT THE MALL

PROJECT VIEW:

A CRM application to manage the services offered by an institution is to provide a centralized, efficient platform for handling interactions, managing services, and optimizing communication between the institution and its stakeholders (such as students, clients, or partners). The goal is to streamline processes, improve user satisfaction, enhance service delivery, and foster long-term relationships with customers or clients. Here's a breakdown of specific aims for such an application

Specific Outcomes:

- Develop an integrated system for tracking vehicle inventory, customer inquiries, sales leads, and follow-ups.

- Implement automated workflows for sales, service, and customer support processes.

- Enable detailed reporting and analytics for performance monitoring and decision-making.

- Ensure seamless integration with other systems like financial and service management tools.

Detailed Steps to Solution Design:

1. Improve Customer Engagement and Experience:

Personalized Communication

Enable tailored communication based on user preferences, past interactions, and service history.

Service Availability and Transparency

Provide users with clear visibility into the services offered, availability, and their own progress (e.g., course registration status, service usage).

Proactive Support

Use data insights to anticipate and address user needs, providing proactive help and ensuring a smoother experience.

2. Enhance Service Management and Delivery:

Streamlined Operations Automate and centralize the management of services, from service requests to fulfillment, ensuring quicker response times and less administrative overhead.

Service Tracking and Reporting Provide administrators and staff with dashboards to monitor the status of services being delivered, track performance against service level agreements (SLAs), and ensure the timely delivery of services.

Efficiency Gains Minimize errors and improve consistency in how services are provided through clear documentation, standard workflows, and automated reminders for both staff and clients.

3. Increase Operational Efficiency and Productivity:

Centralized Data Management Centralize all client data, service information, and communication history in one platform to avoid duplication, reduce errors, and facilitate better decision-making.

Automation of Routine Tasks Automate repetitive tasks like follow-up emails, reminders, and ticket escalations, freeing up staff time to focus on more complex tasks.

Analytics and Reporting Provide detailed insights into user behavior, service performance, and operational bottlenecks to help the institution optimize resources and make data-driven decisions.

4. Enhance Collaboration Across Teams:

Cross-Department Coordination Enable different teams (e.g., admissions, support, sales, marketing) to collaborate and have real-time access to the same data. This ensures consistent communication and reduces fragmentation in service delivery.

Task Management Use features like task assignments, to-do lists, and reminders to keep all team members aligned on key responsibilities and deadlines.

5. Build and Nurture Relationships:

Customer Retention By understanding customer needs, providing excellent service, and anticipating future requirements, a CRM can help institutions retain existing users and build long-term relationships.

Feedback and Improvement Allow users to provide feedback on services, and use that feedback to make continuous improvements, ensuring customer satisfaction and loyalty.

6. Facilitate Data-Driven Decisions:

Performance Metrics Track key performance indicators (KPIs) such as user satisfaction, service delivery time, and engagement levels to assess the success of services.

Market Insights Use CRM analytics to identify trends, spot areas for improvement, and gain insights into market demands, helping the institution adjust its services or marketing strategies accordingly.

7. Scale and Adapt:

Growth Support As the institution grows, a CRM can scale to handle larger volumes of users and services without sacrificing performance or usability.

Customizability A CRM application can be tailored to the specific needs of the institution, whether it's a school, university, healthcare provider, or any other service-oriented organization.

Streamline service management to ensure efficient delivery and tracking of services.

Enhance user engagement through personalized interactions and seamless communication.

Increase operational efficiency by automating tasks and centralizing data.

Facilitate collaboration across different departments involved in service delivery.

Make data-driven decisions through detailed analytics and reporting.

Ensure scalability for future growth and adaptability.

OBJECTIVES:

1. Centralize Customer and Service Data

To consolidate all customer (student, client, or beneficiary) information, service records, and communication history in one centralized system.

Ensure all stakeholders (administrators, staff, and even customers) have easy access to up-to-date and accurate information, enabling better decision-making and reducing data silos.

2. Enhance User Experience

To provide a seamless, personalized, and engaging experience for users (students, clients, or customers).

Use data to offer tailored services, relevant content, and proactive support that anticipates user needs, thereby improving satisfaction and retention.

3. Improve Service Delivery and Efficiency

To automate and streamline the management of services offered by the institution.

Ensure timely and accurate delivery of services by automating workflows, reducing human error, and enhancing internal coordination.

4. Facilitate Communication and Collaboration

To improve communication between the institution and its clients, as well as among internal teams.

Enable easy and consistent communication across various channels (email, SMS, push notifications) and ensure that all departments (sales, marketing, support) are aligned on customer needs and service progress.

5. Track and Monitor Service Performance

To track the progress and performance of services, monitor key metrics, and ensure compliance with service level agreements (SLAs).

Use reporting tools and dashboards to track the timeliness and quality of service delivery, identify areas for improvement, and maintain high standards of service.

6. Increase Customer Retention

To build long-term relationships and ensure high levels of customer satisfaction.

Use insights from CRM data to engage users, follow up on past interactions, and deliver added value that encourages repeat use of services, renewals, or long-term loyalty.

7. Automate Routine Tasks and Administrative Processes

To reduce the manual effort required for administrative tasks such as data entry, follow-up reminders, and scheduling.

Free up staff time for more value-added activities by automating repetitive processes like sending reminders, generating invoices, or updating user profiles.

8. Generate Insights and Analytics

To provide actionable insights into customer behavior, service usage, and operational efficiency.

Use analytics to uncover trends, understand customer preferences, optimize resource allocation, and make data-driven decisions regarding service offerings or marketing strategies.

9. Support Scalability and Growth

To provide a flexible and scalable system that can grow with the institution’s needs.

Ensure the CRM application can handle an increasing number of users, services, and data points as the institution expands, while maintaining performance and usability.

10. Facilitate Lead and Opportunity Management

To capture, track, and nurture leads (e.g., prospective students or clients) through the entire sales or onboarding process.

Improve conversion rates by managing leads efficiently, automating follow-up tasks, and ensuring that sales and support teams have the right information to close opportunities successfully.

anage services, improve customer interactions, and drive overall business success.

 Centralize Data
Provide a single source of truth for all customer and service information.

 Enhance User Experience
Deliver personalized, seamless experiences to customers and users.

 Improve Efficiency
Streamline service management, automate tasks, and improve operational efficiency.

 Facilitate Communication
Enable consistent and effective communication internally and with users.

 Track Performance

Monitor service quality and timeliness to ensure service excellence.

OUTLINE PROCESS OF PROJECT:

1. **Salesforce Reports and Dashboards** help visualize data and track key business metrics for informed decision-making.
2. Reports are customizable, allowing users to filter, group, and summarize data according to specific business needs.
3. Dashboards bring together various reports into a single view, providing real-time insights through visual components like charts and graphs.
4. **Apex Triggers** automate business processes by executing custom logic before or after a record is modified in Salesforce.
5. There are two types of triggers: **Before triggers** for validation and **After triggers** for record updates or related tasks.
6. **Asynchronous Apex** handles long-running tasks in the background, preventing delays in user interactions and system performance.
7. Types of Asynchronous Apex include **Future Methods**, **Queueable Apex**, **Batch Apex**, and **Scheduled Apex**.
8. Data can be imported in bulk using Salesforce's **Data Import Wizard** or **Data Loader**, ensuring seamless integration.
9. Testing is crucial before deployment, including testing in a **Sandbox** and performing **User Acceptance Testing (UAT)**.
10. After deployment, **training, documentation**, and **ongoing support** are key to maintaining the system's effectiveness and improving it over time.

1. Centralize Customer and Service Data:

Actions

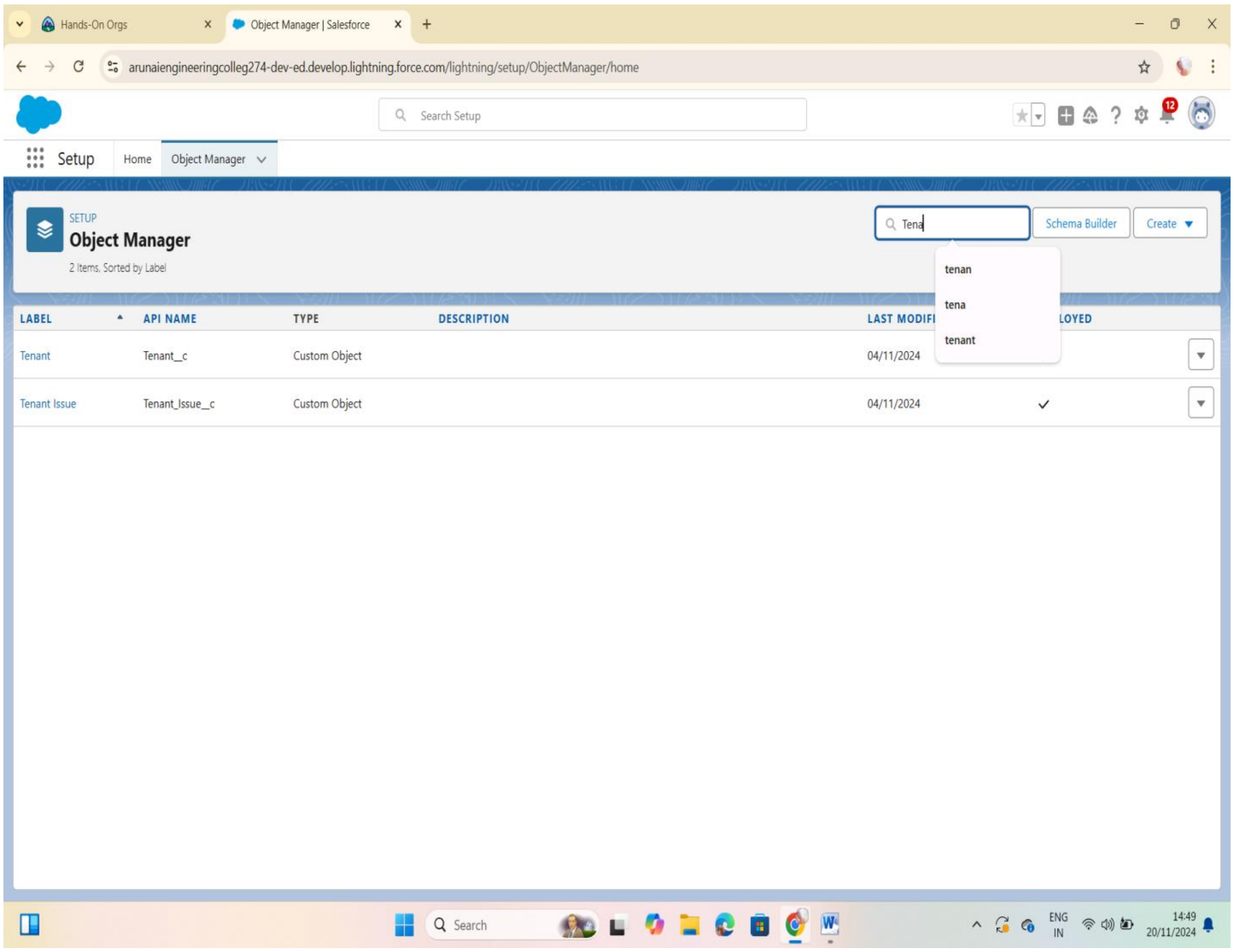
- Unified Database Create a central database where all client and service information (demographics, communication history, service requests, payment details) is stored.
- Integration Integrate the CRM with other institutional systems (e.g., learning management systems (LMS), payment gateways, student management systems) to automatically pull in relevant data.
- Customer Profiles Build detailed, up-to-date profiles for each client, which include personal information, service history, and interaction history (e.g., emails, calls, support tickets).

Expected Outcome

- A comprehensive, 360-degree view of each customer and service provided by the institution.
- Reduced risk of data silos, ensuring that all departments (e.g., admissions, support, marketing) can access consistent information.

1. Create Object from Tenant:

- **Purpose**: To import data from an external source (like an Excel spreadsheet) into Salesforce and create a new custom object based on that data.
- **Steps**:
  - Prepare the spreadsheet with the relevant data you want to import (e.g., student records, property details, etc.).
  - In Salesforce ,navigate to Object manager → **Create a Tenant Object→ Create a Lease Tracking Object → Create a Tenant issue Object**
  - Define the custom object's name and fields based on the columns in your spreadsheet.
  - Use **Data Import Wizard** or **Data Loader** to upload the data from the spreadsheet into Salesforce.
  - Outcome A custom Salesforce object that can store data from the spreadsheet, with fields and records populated automatically.


SETUP

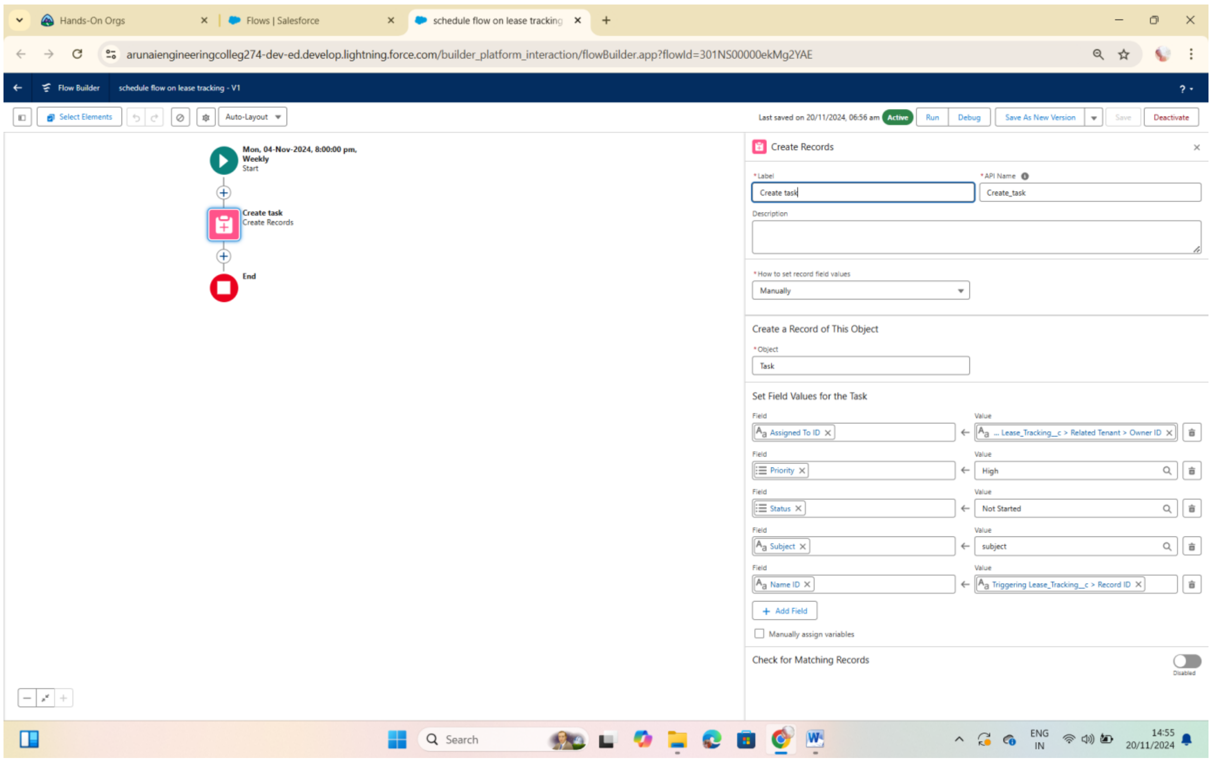
Object Manager

2 Items, Sorted by Label

LABEL	API NAME	TYPE	DESCRIPTION	LAST MODIFIED	DEPLOYED
Tenant	Tenant__c	Custom Object		04/11/2024	
Tenant Issue	Tenant_Issue__c	Custom Object		04/11/2024	✓

2. Create a Screen Flow for CRM Application Management To The Mall

- **Purpose**: To build a user-friendly, guided application process for student admissions


Flow Builder — schedule flow on lease tracking - V1

Last saved on 20/11/2024, 06:56 am Active

Mon, 04-Nov-2024, 8:00:00 pm, Weekly — Start

Create task — Create Records

End

Create Records

*Label: Create task

*API Name: Create_task

Description:

*How to set record field values: Manually

Create a Record of This Object

*Object: Task

Set Field Values for the Task

Field	Value
Assigned To ID	... Lease_Tracking__c > Related Tenant > Owner ID
Priority	High
Status	Not Started
Subject	subject
Name ID	Triggering Lease_Tracking__c > Record ID

☐ Manually assign variables

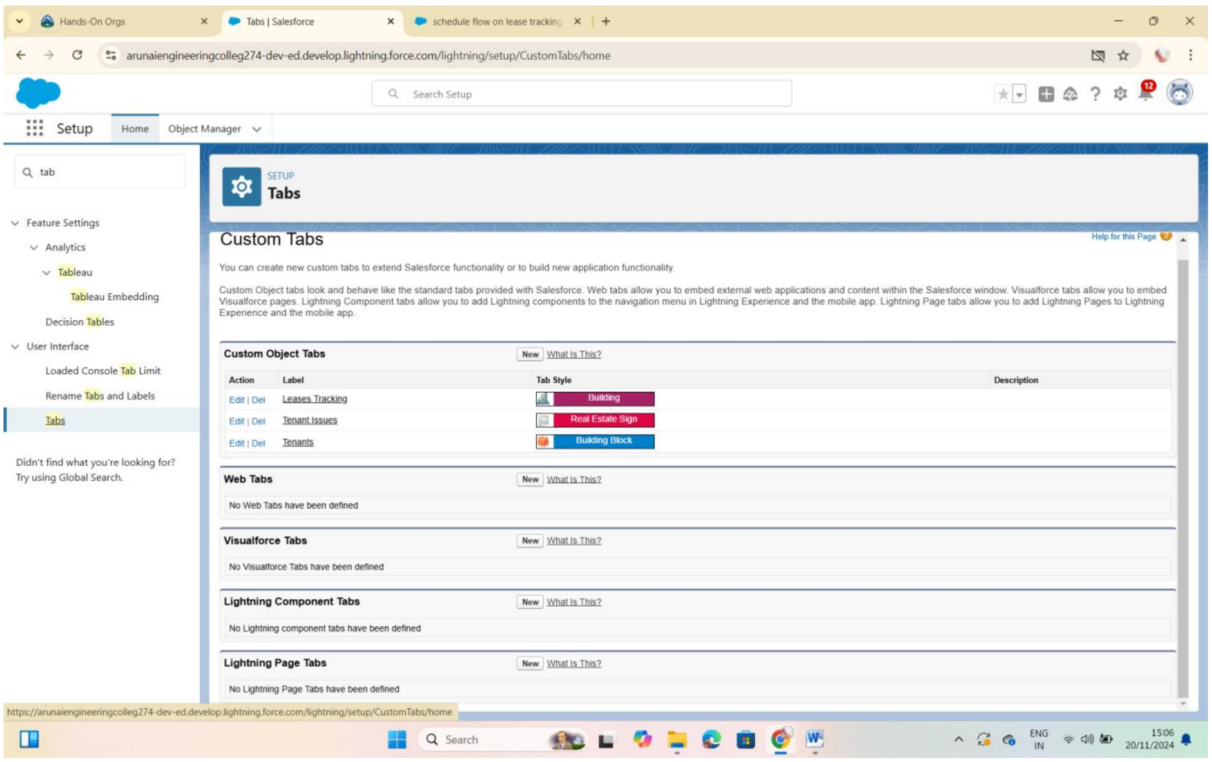
Check for Matching Records: Disabled

using Salesforce Flow (Screen Flow).

- **Steps**:
    - In Salesforce, navigate to **Setup** → **Flow** → **Create New Flow** → **Screen Flow**.
        - **Record Triggered Flow** on the **Tenant** object.
    - **Trigger Condition**: Only when the **GST No** field is null on creation.
    - **Action**: Send an email to the tenant requesting their GST No.
    - **Save and Activate** the flow..

3.Create Tabs:

- **Standard Tabs**: Predefined tabs for standard Salesforce objects (e.g., Account, Contact).
- **Custom Tabs**: Created for custom objects to manage your organization's specific data.
- **Web Tabs**: Used to display external websites or applications within Salesforce.
- **Visualforce Tabs**: Used to display custom pages created with Visualforce (and Apex) to provide custom functionality.
- Each type of tab serves different purposes, allowing you to tailor the Salesforce interface to your organization's needs, from accessing standard objects to integrating external resources and custom-built solutions.



Custom Tabs

You can create new custom tabs to extend Salesforce functionality or to build new application functionality.

Custom Object tabs look and behave like the standard tabs provided with Salesforce. Web tabs allow you to embed external web applications and content within the Salesforce window. Visualforce tabs allow you to embed Visualforce pages. Lightning Component tabs allow you to add Lightning components to the navigation menu in Lightning Experience and the mobile app. Lightning Page tabs allow you to add Lightning Pages to Lightning Experience and the mobile app.

Custom Object Tabs New What Is This?

Action	Label	Tab Style	Description
Edit | Del	Leases Tracking	Building	
Edit | Del	Tenant Issues	Real Estate Sign	
Edit | Del	Tenants	Building Block	

Web Tabs New What Is This?

No Web Tabs have been defined

Visualforce Tabs New What Is This?

No Visualforce Tabs have been defined

Lightning Component Tabs New What Is This?

No Lightning component tabs have been defined

Lightning Page Tabs New What Is This?

No Lightning Page Tabs have been defined

https://arunaiengineeringcolleg274-dev-ed.develop.lightning.force.com/lightning/setup/CustomTabs/home

4. Create Fields and Relationships:

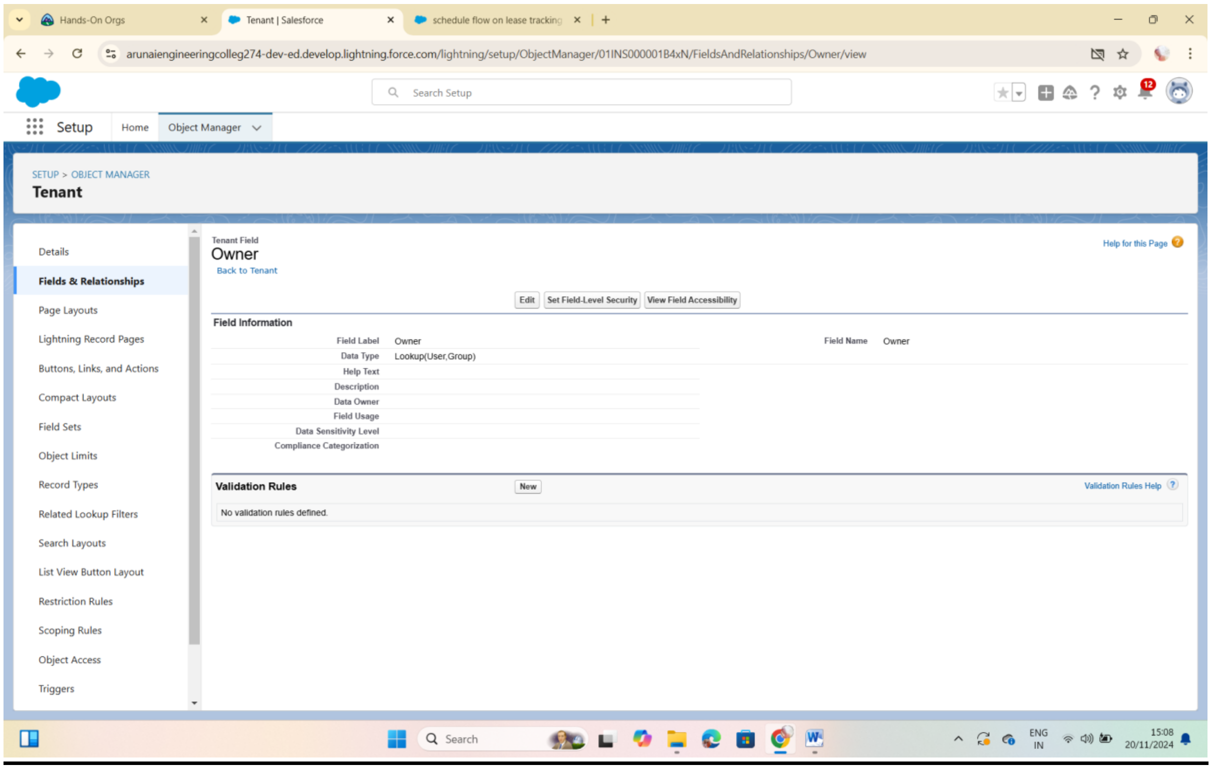
Creating Fields:

1. **Fields** capture specific data for an object (e.g., Tenant Name, Rent Amount).
2. To create a field:
   - Go to **Setup** → **Object Manager** → Select the object → **Fields & Relationships** → **New**.
   - Choose the **field type** (e.g., Text, Number, Picklist, Currency).
   - Set the field's properties (label, length, required status).
   - Configure **field-level security** and add it to **page layouts**.
3. Salesforce provides various field types for different data, including:
   - **Text**, **Number**, **Currency**, **Picklist**, **Formula**, **Checkbox**, etc.

Creating Relationships:

1. **Relationships** define how objects are connected in Salesforce. They ensure that records in one object are related to records in another object.
2. Types of relationships:
   - **Lookup Relationship**: A one-to-one relationship where the child record can exist independently of the parent.
   - **Master-Detail Relationship**: A stronger relationship where the child record's existence depends on the parent. If the parent is deleted, the child is also deleted.

- **Many-to-Many Relationship (Junction Object)**: Uses two **Master-Detail relationships** to create a many-to-many connection between two objects (e.g., linking Students to Courses).
- **External Lookup/Indirect Lookup**: Links Salesforce records to external objects or external data sources.



Tenant Field
Owner
Back to Tenant

Field Information

Field Label	Owner	Field Name	Owner
Data Type	Lookup(User,Group)		
Help Text			
Description			
Data Owner			
Field Usage			
Data Sensitivity Level			
Compliance Categorization			

Validation Rules

No validation rules defined.

5.Create the Lightning App:

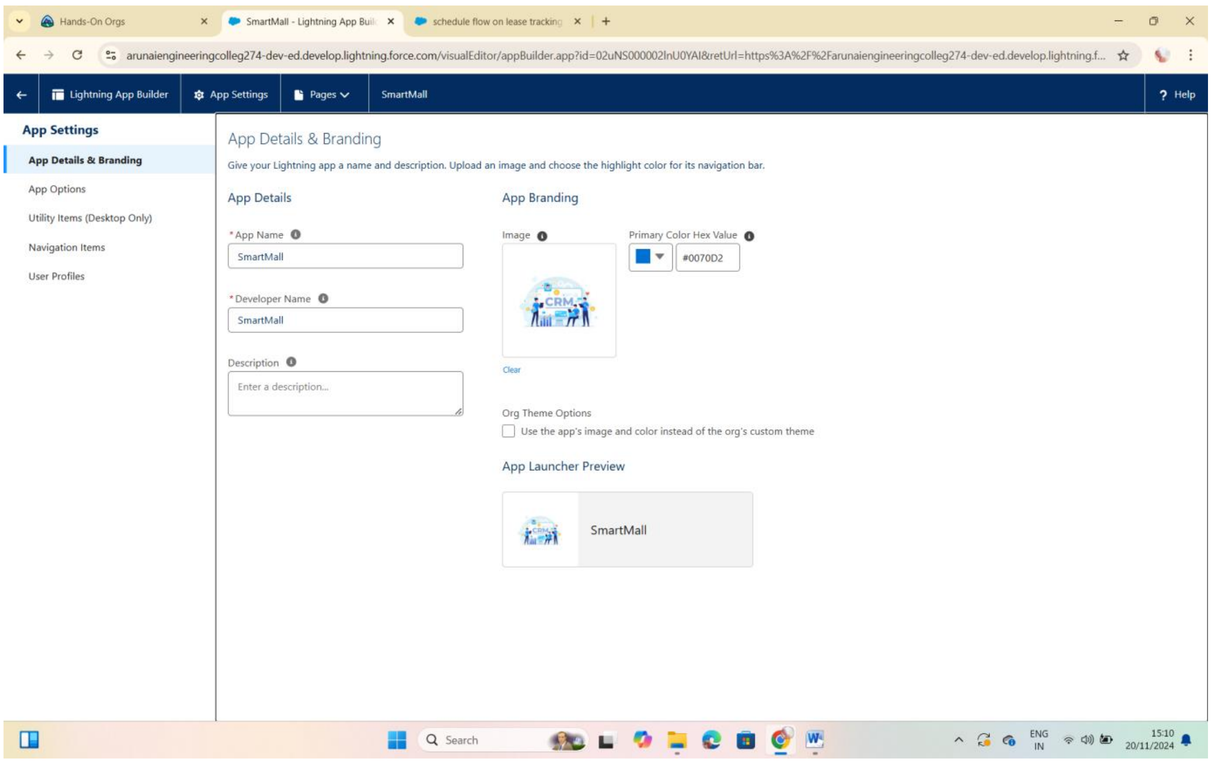
A **Lightning App** in Salesforce is a customized collection of tabs, objects, pages, and tools, all bundled together for a specific business function. It improves user efficiency by providing easy access to key resources in the navigation bar.

Types of Salesforce Apps:

1. **Standard Apps**: Default apps (e.g., Sales, Marketing, Chatter) that come with Salesforce. You cannot modify their logos or labels.
2. **Custom Apps**: Apps created to fit your organization's specific needs. These can include custom tabs, logos, and branding.

Steps to Create a Custom Lightning App:

1. **Navigate to Setup → App Manager → New Lightning App**.
2. **Define App Details**:
   - Set the app's name, description, and branding (logo and color).



arunaiengineeringcolleg274-dev-ed.develop.lightning.force.com/visualEditor/appBuilder.app?id=02uNS000002InU0YAI&retUrl=https%3A%2F%2Farunaiengineeringcolleg274-dev-ed.develop.lightning.f...

App Settings

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 Name: SmartMall

*Developer Name: SmartMall

Description: Enter a description...

App Branding

Image

Primary Color Hex Value: #0070D2

Org Theme Options

☐ Use the app's image and color instead of the org's custom theme

App Launcher Preview

SmartMall

15:10 20/11/2024

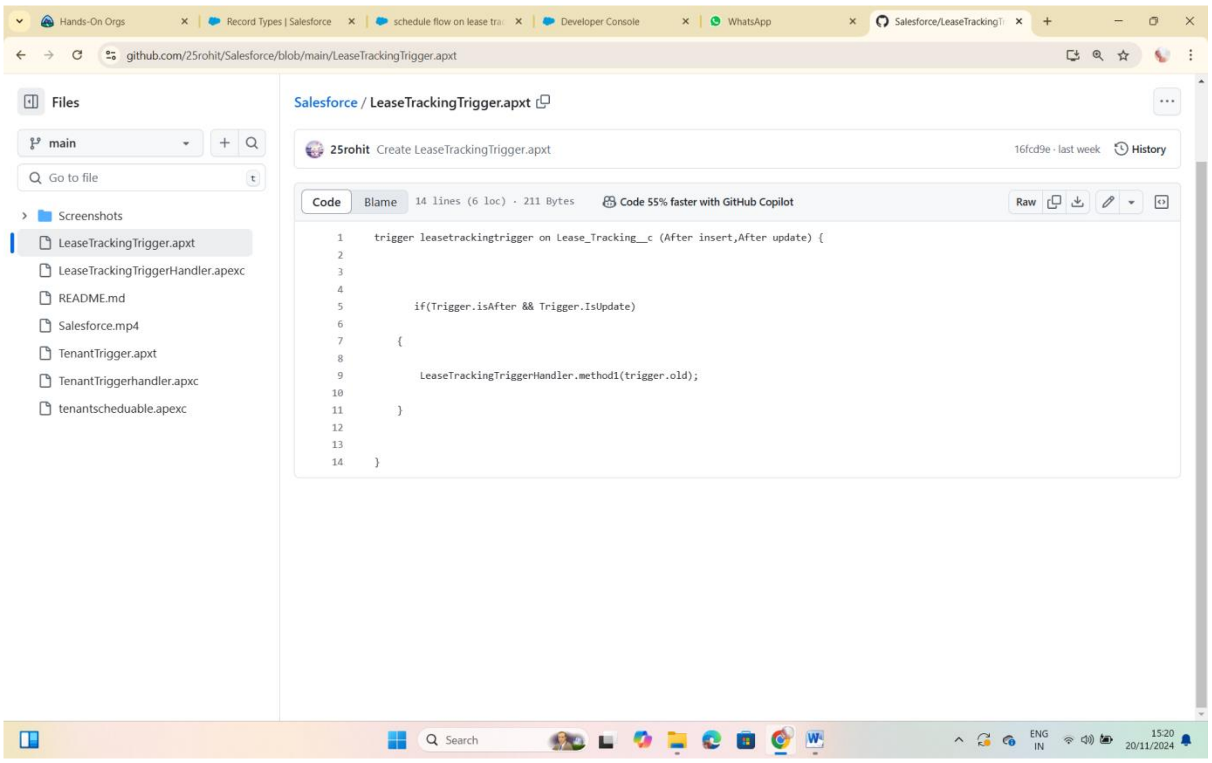
3. **Choose Navigation Style**:
   - **Standard** or **Console** navigation, based on app use case.
4. **Add Items to App**:
   - Select the tabs (standard or custom objects), pages, and tools that users will need.
5. **Configure User Access**:
   - Control who can access the app based on user profiles or permission sets.
6. **Add Utility Bar (Optional)**:
   - Include quick-access tools like notes, tasks, or custom components.
7. **Customize Pages (Optional)**:
   - Use **App Builder** to design Lightning Pages for better UI/UX.
8. **Save, Assign, and Test**:
   - Save the app, assign it to the right users, and test it for functionality.

6. Apex Triggers:

Apex Triggers are pieces of Apex code that execute automatically before or after **DML (Data Manipulation Language)** operations (like insert, update, delete) on Salesforce records. They help automate tasks, enforce business rules, and modify data that would be difficult to handle using only the standard Salesforce UI.

Types of Apex Triggers:

1. **Before Triggers**:
   - Execute **before** changes are committed to the database.
   - Used for data validation or modifications before saving records.
2. **After Triggers**:
   - Execute **after** changes are committed to the database.
   - Used for tasks like updating related records or sending notifications.



github.com/25rohit/Salesforce/blob/main/LeaseTrackingTrigger.apxt

Salesforce / LeaseTrackingTrigger.apxt

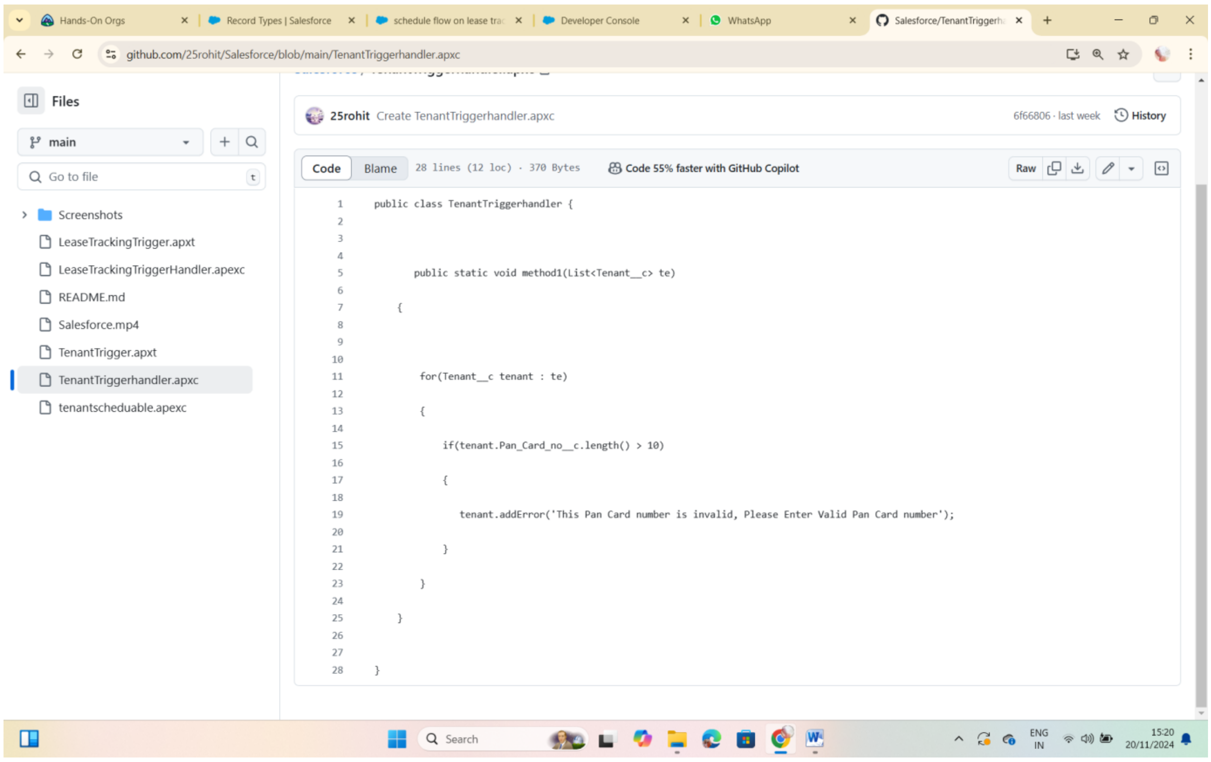
25rohit Create LeaseTrackingTrigger.apxt

14 lines (6 loc) · 211 Bytes

```
trigger leasetrackingtrigger on Lease_Tracking__c (After insert,After update) {



      if(Trigger.isAfter && Trigger.IsUpdate)

    {

       LeaseTrackingTriggerHandler.method1(trigger.old);

    }


}
```

How to Create a Trigger:

1. Open the **Developer Console** in Salesforce.
2. Go to the **File** menu and select **New → Apex Trigger**.
3. Provide the **trigger name** and select the object to trigger the code.
4. Write the trigger logic in the code editor.

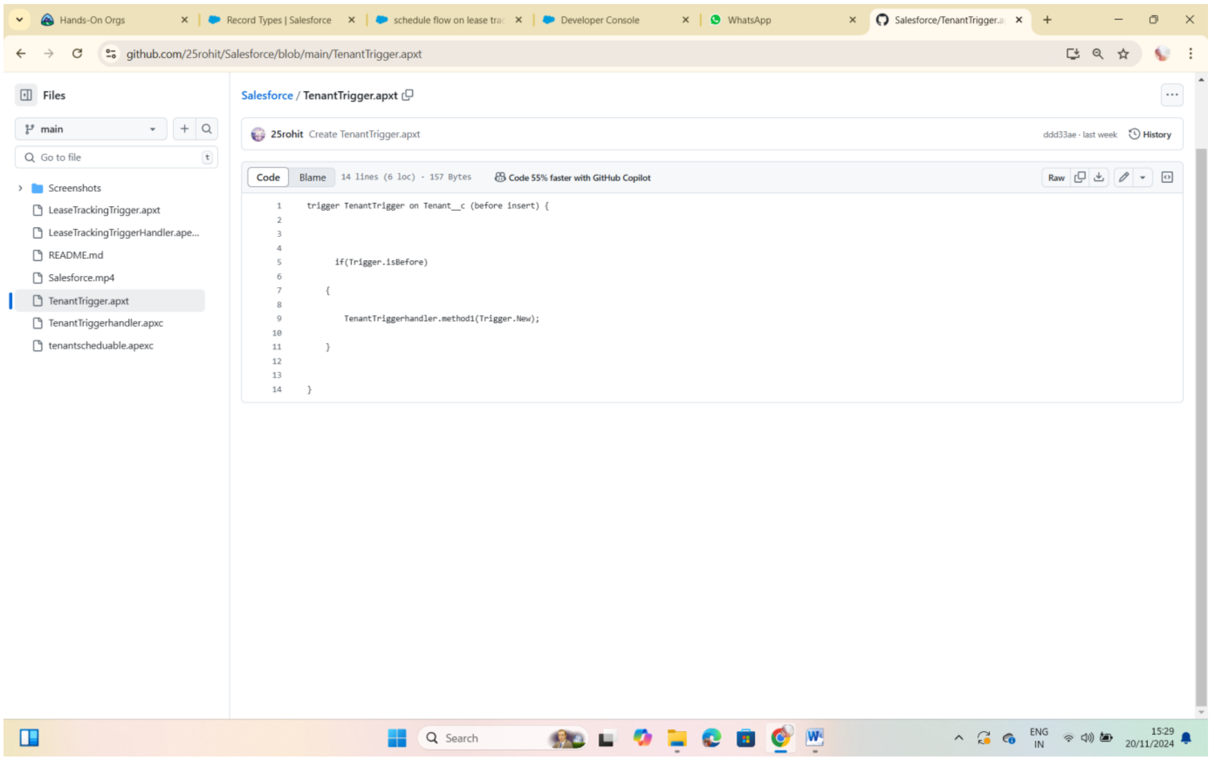


github.com/25rohit/Salesforce/blob/main/TenantTriggerhandler.apxc

25rohit Create TenantTriggerhandler.apxc

28 lines (12 loc) · 370 Bytes

```
public class TenantTriggerhandler {



      public static void method1(List<Tenant__c> te)

    {



       for(Tenant__c tenant : te)

       {

          if(tenant.Pan_Card_no__c.length() > 10)

          {

            tenant.addError('This Pan Card number is invalid, Please Enter Valid Pan Card number');

          }

       }

    }


}
```



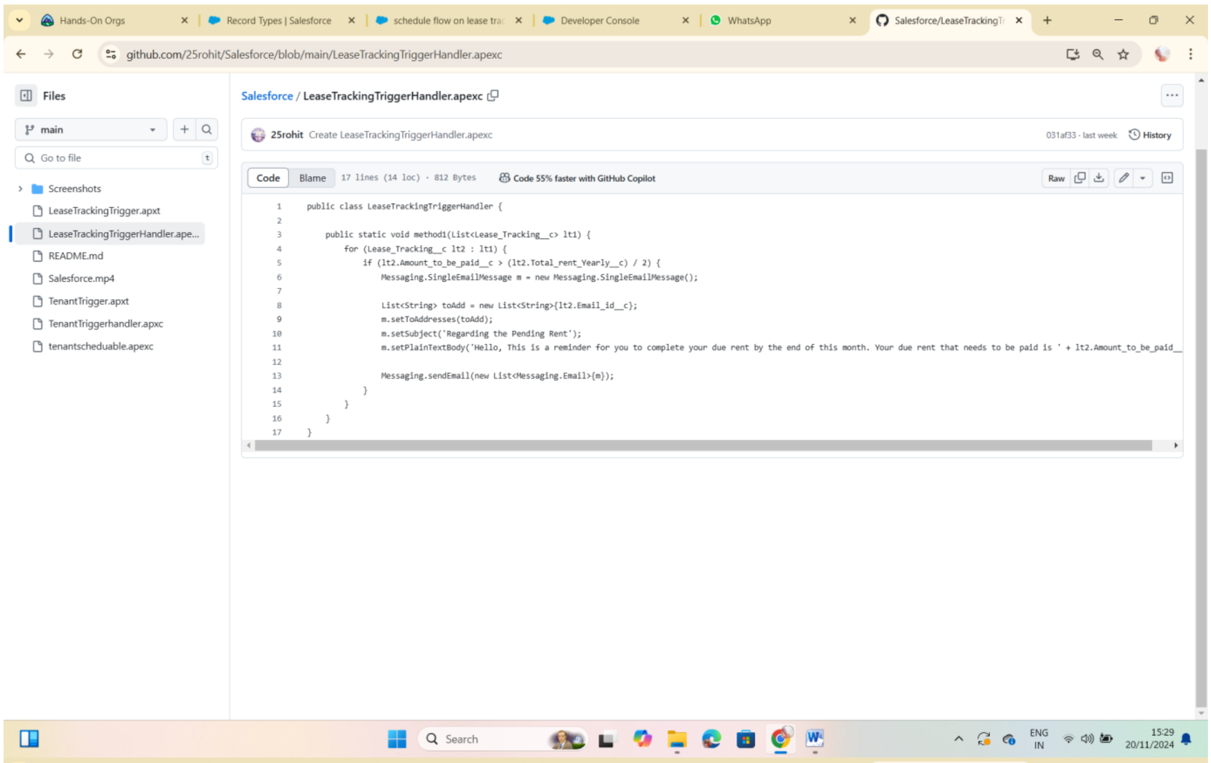
github.com/25rohit/Salesforce/blob/main/TenantTrigger.apxt

Salesforce / TenantTrigger.apxt

25rohit Create TenantTrigger.apxt

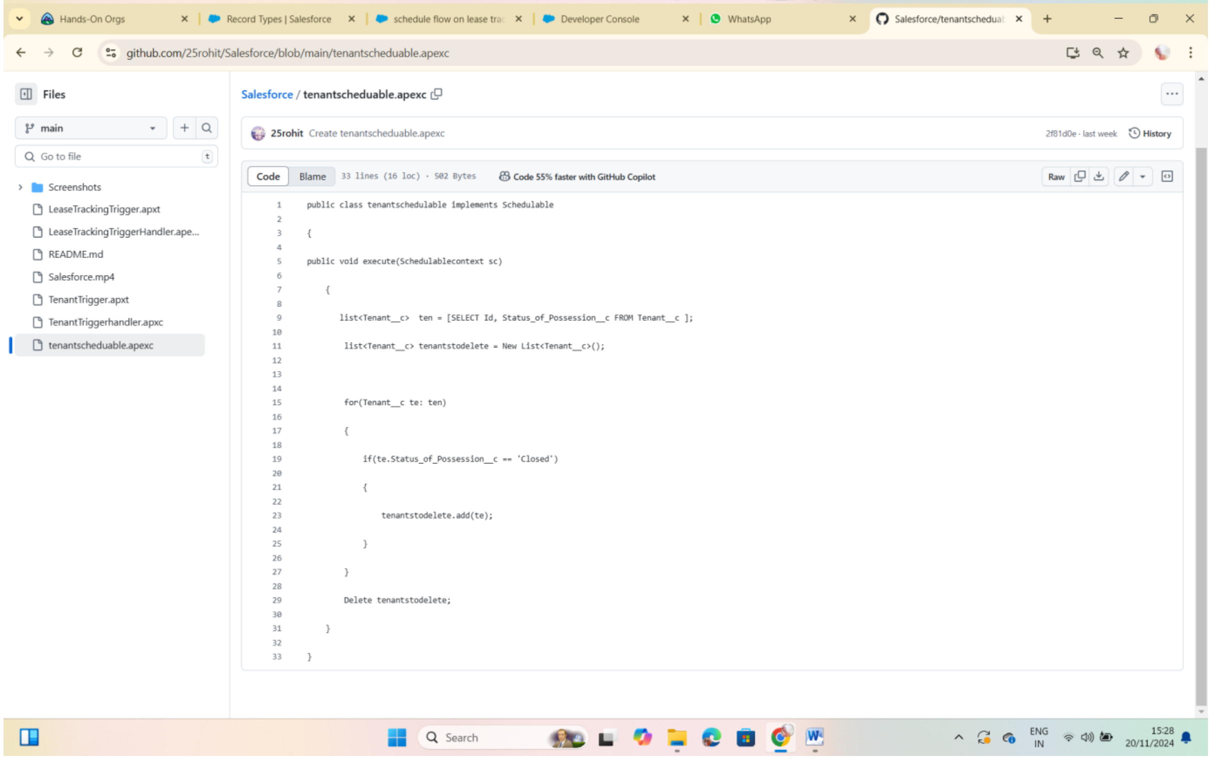
14 lines (6 loc) · 157 Bytes

```
trigger TenantTrigger on Tenant__c (before insert) {



    if(Trigger.isBefore)

   {

     TenantTriggerhandler.method1(Trigger.New);

   }


}
```


Salesforce / LeaseTrackingTriggerHandler.apexc

25rohit Create LeaseTrackingTriggerHandler.apexc

```
public class LeaseTrackingTriggerHandler {

    public static void method1(List<Lease_Tracking__c> lt1) {
        for (Lease_Tracking__c lt2 : lt1) {
            if (lt2.Amount_to_be_paid__c > (lt2.Total_rent_Yearly__c) / 2) {
                Messaging.SingleEmailMessage m = new Messaging.SingleEmailMessage();

                List<String> toAdd = new List<String>{lt2.Email_id__c};
                m.setToAddresses(toAdd);
                m.setSubject('Regarding the Pending Rent');
                m.setPlainTextBody('Hello, This is a reminder for you to complete your due rent by the end of this month. Your due rent that needs to be paid is ' + lt2.Amount_to_be_paid__

                Messaging.sendEmail(new List<Messaging.Email>{m});
            }
        }
    }
}
```

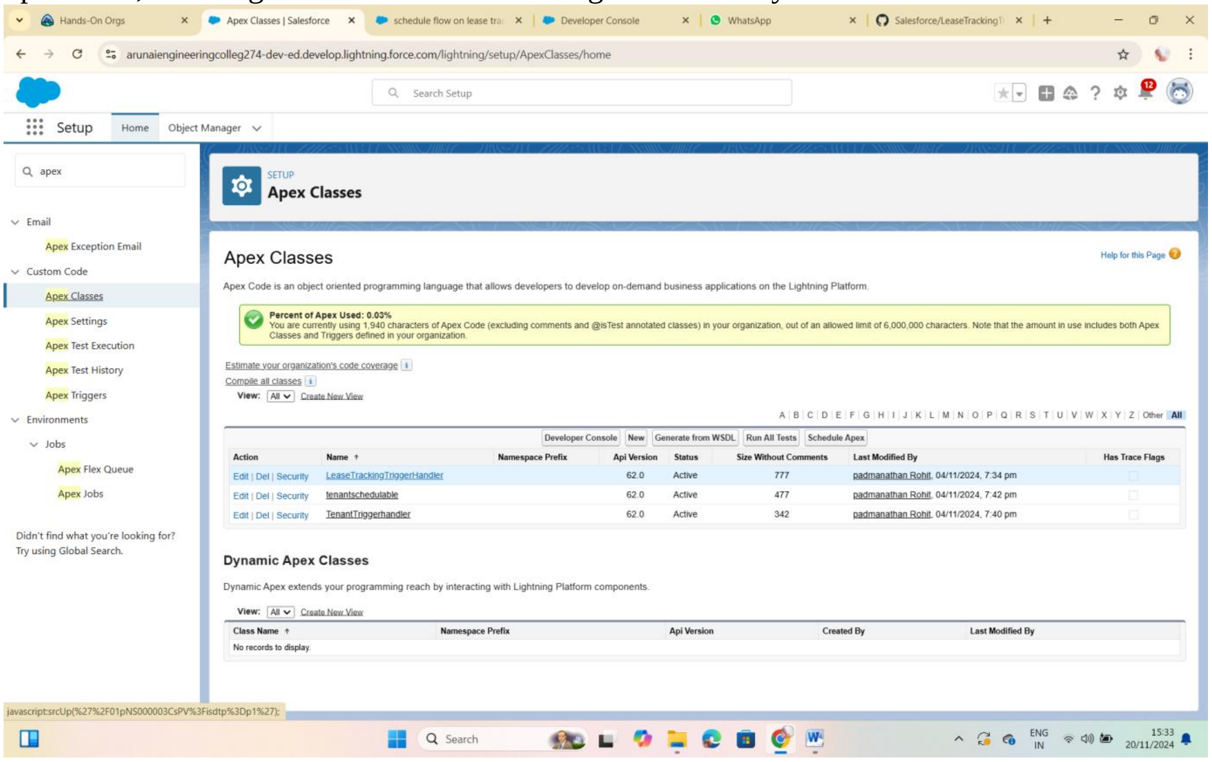

Salesforce / tenantscheduable.apexc

25rohit Create tenantscheduable.apexc

```
public class tenantschedulable implements Schedulable

{

public void execute(Schedulablecontext sc)

    {

       list<Tenant__c>  ten = [SELECT Id, Status_of_Possession__c FROM Tenant__c ];

        list<Tenant__c> tenantstodelete = New List<Tenant__c>();



        for(Tenant__c te: ten)

        {

            if(te.Status_of_Possession__c == 'Closed')

            {

                tenantstodelete.add(te);

            }

        }

        Delete tenantstodelete;

    }

}
```

7. Asynchronous Apex:

Asynchronous Apex in Salesforce is a programming model where code runs independently in the background, separate from the user's immediate context. It is used to handle long-running processes, complex calculations, or tasks that could otherwise slow down or block user interactions. This approach improves performance by offloading resource-intensive operations, allowing users to continue working without delays.


SETUP
Apex Classes

Apex Classes

Apex Code is an object oriented programming language that allows developers to develop on-demand business applications on the Lightning Platform.

Percent of Apex Used: 0.03%
You are currently using 1,940 characters of Apex Code (excluding comments and @isTest annotated classes) in your organization, out of an allowed limit of 6,000,000 characters. Note that the amount in use includes both Apex Classes and Triggers defined in your organization.

Estimate your organization's code coverage
Compile all classes
View: All Create New View

Action	Name ↑	Namespace Prefix	Api Version	Status	Size Without Comments	Last Modified By	Has Trace Flags
Edit | Del | Security	LeaseTrackingTriggerHandler		62.0	Active	777	padmanathan Rohit, 04/11/2024, 7:34 pm	
Edit | Del | Security	tenantschedulable		62.0	Active	477	padmanathan Rohit, 04/11/2024, 7:42 pm	
Edit | Del | Security	TenantTriggerhandler		62.0	Active	342	padmanathan Rohit, 04/11/2024, 7:40 pm	

Dynamic Apex Classes

Dynamic Apex extends your programming reach by interacting with Lightning Platform components.

View: All Create New View

Class Name ↑	Namespace Prefix	Api Version	Created By	Last Modified By
No records to display.				

8. Create Reports And Dashboards

Reports and **Dashboards** in Salesforce are tools that help users visualize and analyze data for better decision-making and performance tracking.

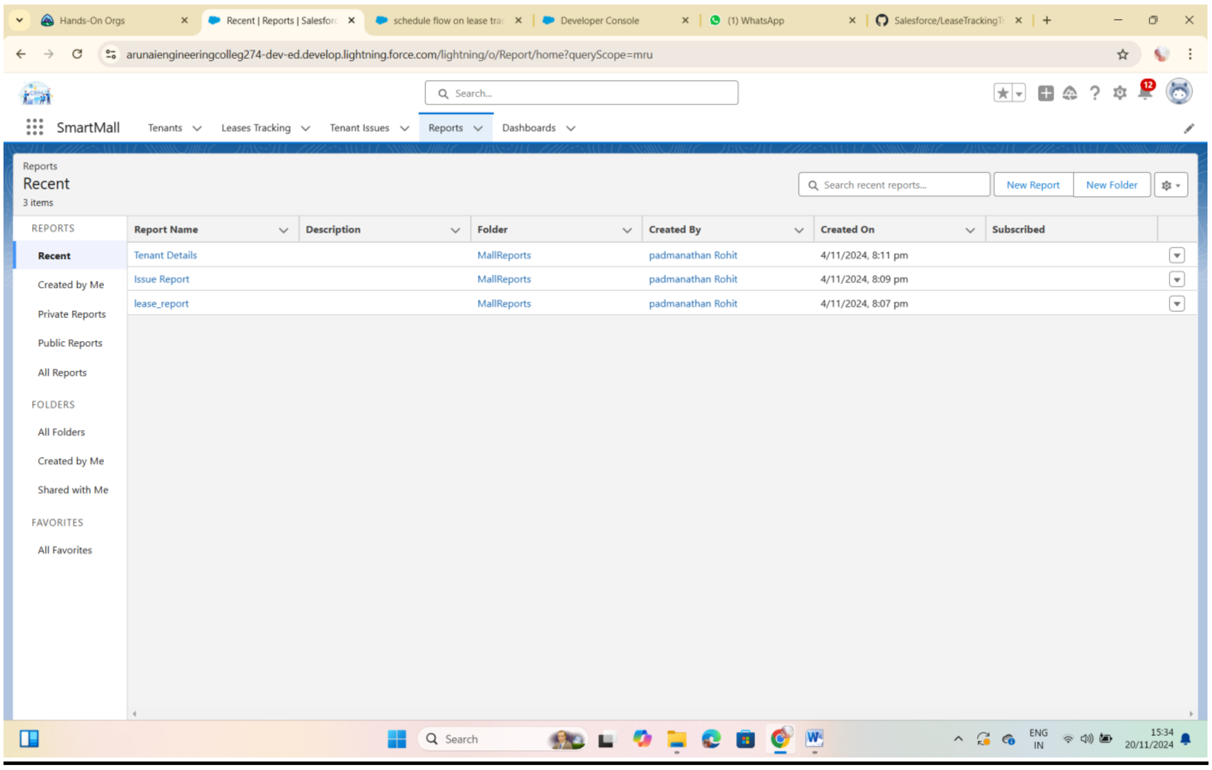
- **Reports**: Organized data views that can be customized to track key metrics. Types include:
  - **Tabular**: Simple lists of data.
  - **Summary**: Grouped data with subtotals.
  - **Matrix**: Data organized in rows and columns.
  - **Joined**: Combines multiple report blocks.
- **Dashboards**: Visual representations of key metrics using components like charts and tables. Dashboards aggregate data from reports to provide a comprehensive overview of business performance.

Creating Reports:

1. Go to the **Reports** tab, create a new report, select an object, and customize filters, groupings, and summaries.

Creating Dashboards:

1. Go to the **Dashboards** tab, create a new dashboard, and add components based on reports.

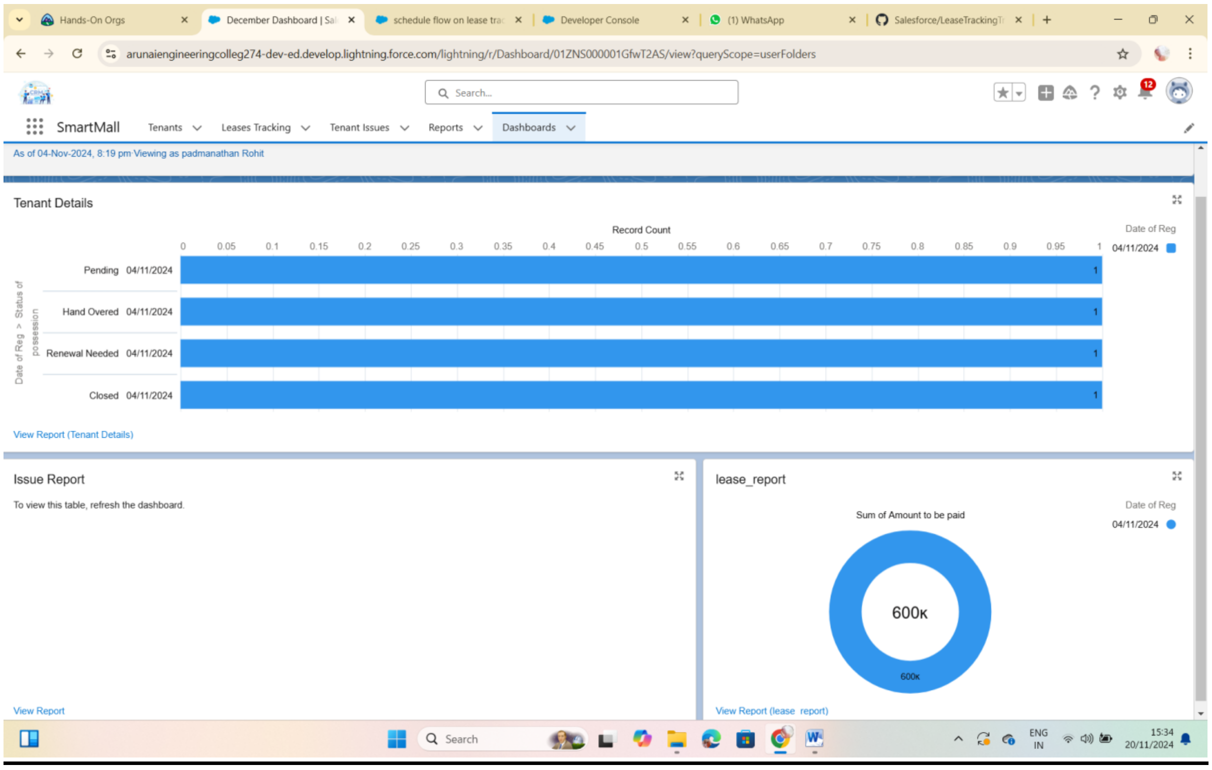


arunaiengineeringcolleg274-dev-ed.develop.lightning.force.com/lightning/o/Report/home?queryScope=mru

SmartMall   Tenants   Leases Tracking   Tenant Issues   Reports   Dashboards

Reports
Recent
3 items

Report Name	Description	Folder	Created By	Created On	Subscribed
Tenant Details		MallReports	padmanathan Rohit	4/11/2024, 8:11 pm	
Issue Report		MallReports	padmanathan Rohit	4/11/2024, 8:09 pm	
lease_report		MallReports	padmanathan Rohit	4/11/2024, 8:07 pm	



arunaiengineeringcolleg274-dev-ed.develop.lightning.force.com/lightning/r/Dashboard/01ZNS000001GfwT2AS/view?queryScope=userFolders

As of 04-Nov-2024, 8:19 pm Viewing as padmanathan Rohit

Tenant Details

Date of Reg > Status of possession	Record Count
Pending 04/11/2024	1
Hand Overed 04/11/2024	1
Renewal Needed 04/11/2024	1
Closed 04/11/2024	1

View Report (Tenant Details)

Issue Report

To view this table, refresh the dashboard.

View Report

lease_report

Sum of Amount to be paid: 600K

View Report (lease_report)

Key Steps:

 Reports in Salesforce: Organize and display data in formats like Tabular, Summary, Matrix, or Joined to track business performance and KPIs.

 Dashboards: Visualize data through charts and graphs, aggregating information from reports to provide an interactive, real-time view of business performance.

 Creating Reports: Use the **Reports** tab to create customized reports by selecting the object, applying filters, and grouping data to meet your analysis needs.

 Creating Dashboards: In the **Dashboards** tab, combine multiple report components (e.g., charts, tables) into one view for a comprehensive analysis.

 Apex Triggers: Automate actions in Salesforce by writing triggers that run **before** or **after** DML operations (insert, update, delete) on records.

 Asynchronous Apex: Use **Future Methods**, **Queueable Apex**, **Batch Apex**, and **Scheduled Apex** to handle long-running or complex tasks in the background without blocking user actions.

 Inserting Records: Add records manually, use the **Data Import Wizard** for bulk imports, or insert records programmatically using **Apex**.

 Data Import Tools: Use the **Data Import Wizard** for straightforward CSV imports or **Data Loader** for more complex, large-scale data operations.

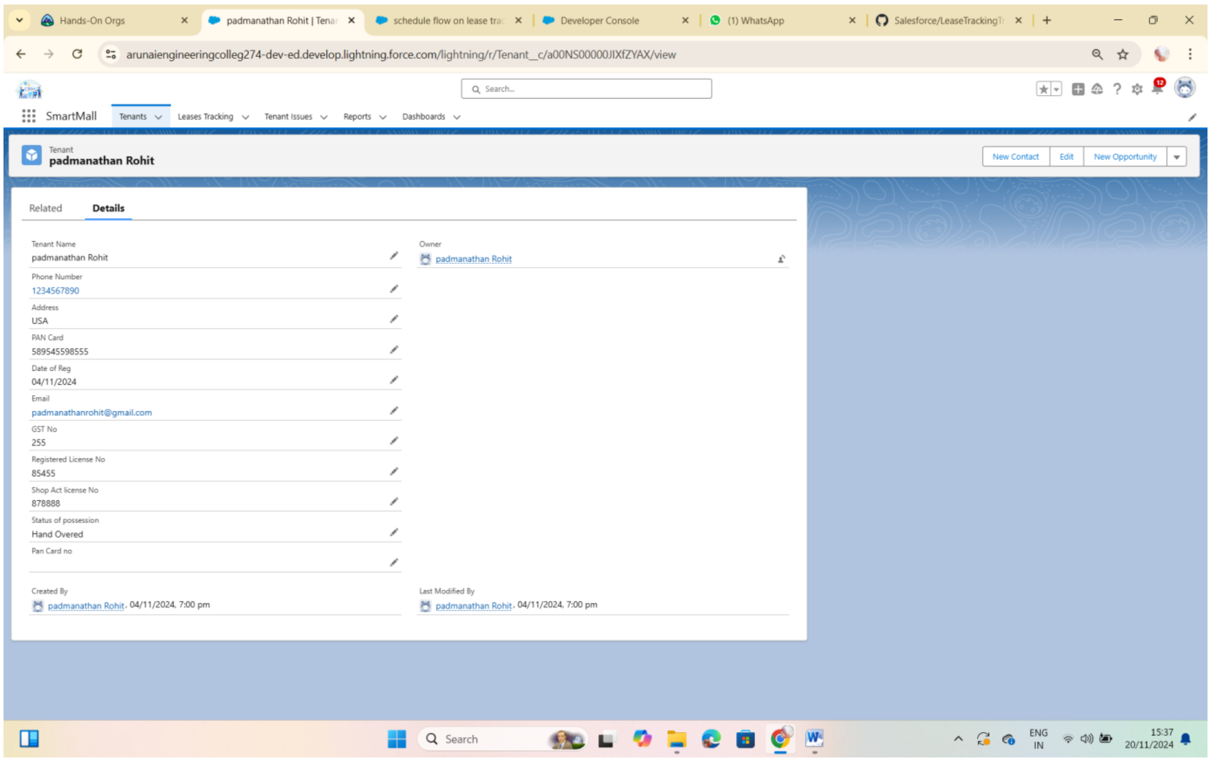
 Trigger Logic: Define logic in Apex triggers to perform validations, calculations, or related record updates automatically during data changes.

 Real-Time Reporting: Dashboards allow real-time visualization of key metrics, enabling data-driven decision-making across teams.

Project process:

- **Define Requirements**:
  - Identify business goals.
  - Select features: Reports, Dashboards, Triggers, and Asynchronous Apex.
  - Identify users and stakeholders.
- **Create Reports**:
  - Navigate to **Reports** tab and create new reports.
  - Apply filters and groupings as per requirements.
  - Save and test the report outputs.
- **Create Dashboards**:

    - Navigate to **Dashboards** tab.
    - Add report components like charts and tables.
    - Customize the layout and share with the team.
- **Develop Apex Triggers**:
    - Identify automation scenarios (e.g., Opportunity updates).
    - Write and test Apex triggers for data changes.
    - Deploy after successful testing.
- **Implement Asynchronous Apex**:
    - Identify tasks requiring background processing (e.g., large data processing).
    - Use **Future Methods**, **Queueable**, or **Batch Apex**.
    - Test the functionality in **Sandbox** before deploying.
- **Testing**:
    - Import any necessary data via **Data Import Wizard** or **Data Loader**.
    - Perform unit tests and User Acceptance Testing (UAT).
- **Deployment**:
    - Deploy the solution to **Production** using Change Sets or Salesforce CLI.
    - Monitor the system for any errors or performance issues.
- **Training & Documentation**:
    - Train end-users on how to use reports, dashboards, and triggers.
    - Provide documentation for ongoing maintenance.
- **Ongoing Support**:
    - Regularly monitor reports and dashboards.
    - Address any issues with asynchronous processes.
- **Continuous Improvement**:
    - Collect feedback from users for further enhancement.
    - Iteratively improve the solution to meet evolving needs.



arunaiengineeringcolleg274-dev-ed.develop.lightning.force.com/lightning/r/Tenant__c/a00NS00000JIXfZYAX/view
SmartMall
Tenants
Leases Tracking
Tenant Issues
Reports
Dashboards
New Contact
Edit
New Opportunity
Related
Details

Tenant
padmanathan Rohit

Tenant Name	padmanathan Rohit
Phone Number	1234567890
Address	USA
PAN Card	589545598555
Date of Reg	04/11/2024
Email	padmanathanrohit@gmail.com
GST No	255
Registered License No	85455
Shop Act license No	878888
Status of possession	Hand Overed
Pan Card no	
Owner	padmanathan Rohit
Created By	padmanathan Rohit, 04/11/2024, 7:00 pm
Last Modified By	padmanathan Rohit, 04/11/2024, 7:00 pm

CONCLUSION:

Implementing Salesforce features such as **Reports**, **Dashboards**, **Apex Triggers**, and **Asynchronous Apex** is crucial for organizations looking to optimize their business processes, automate tasks, and gain real-time insights into their data. By following a structured project process, businesses can effectively design, implement, and deploy these features to enhance user productivity and make informed decisions.

Key takeaways from the process include:

- **Reports and Dashboards** allow organizations to visualize data and track performance metrics, enabling data-driven decision-making.
- **Apex Triggers** provide automation capabilities that allow businesses to streamline workflows, enforce data validation, and handle complex data operations.
- **Asynchronous Apex** helps with long-running or resource-intensive processes, improving system performance and preventing user interruptions.
- Thorough testing, proper deployment, user training, and ongoing support are essential to ensure the successful implementation of these features.