Phase 2: Org Setup & Configuration

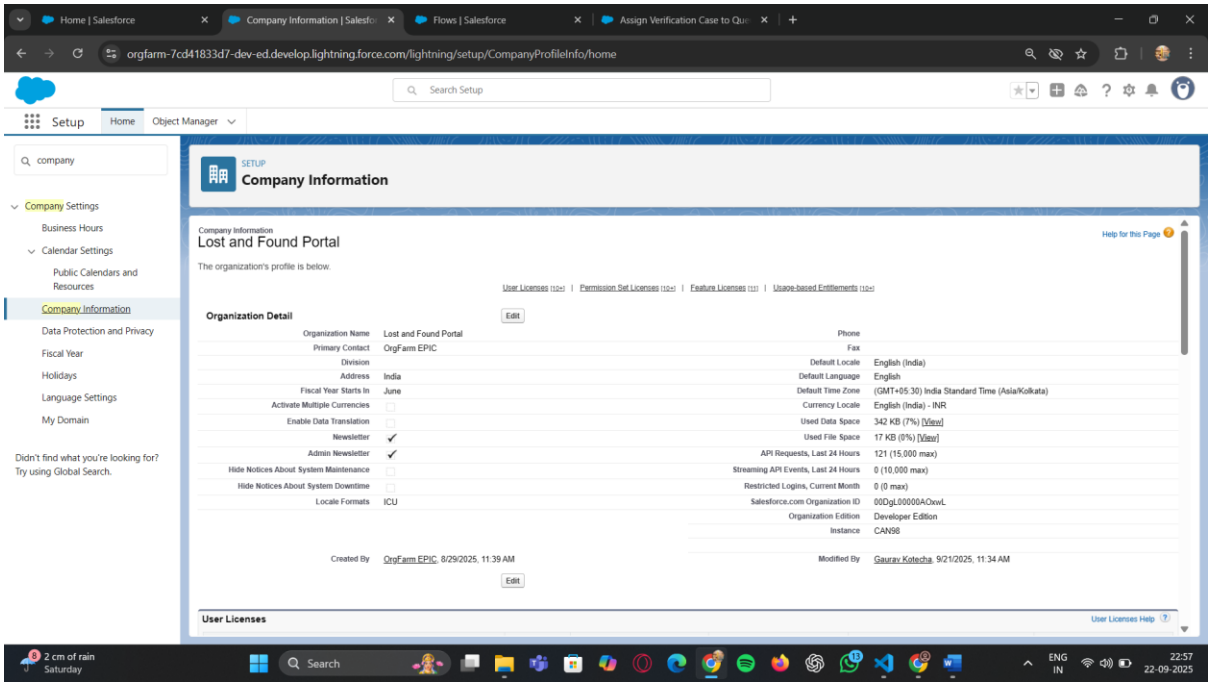
2.1 Introduction

The objective of Phase 2 was to establish the foundational Salesforce environment for the Lost and Found Portal. This involved setting up the company profile, creating a robust security and sharing model, and implementing the core automation to assign incoming cases to the appropriate team.

2.2 Company Profile & Org Defaults

The organization's basic details and default settings were configured to align with the project's operational context.

- **Salesforce Edition:** Enterprise Edition
- **Company Name:** Campus Lost & Found Services Pvt. Ltd.
- **Default Locale:** English (India)
- **Default Time Zone:** Asia/Kolkata (IST)
- **Default Currency:** INR

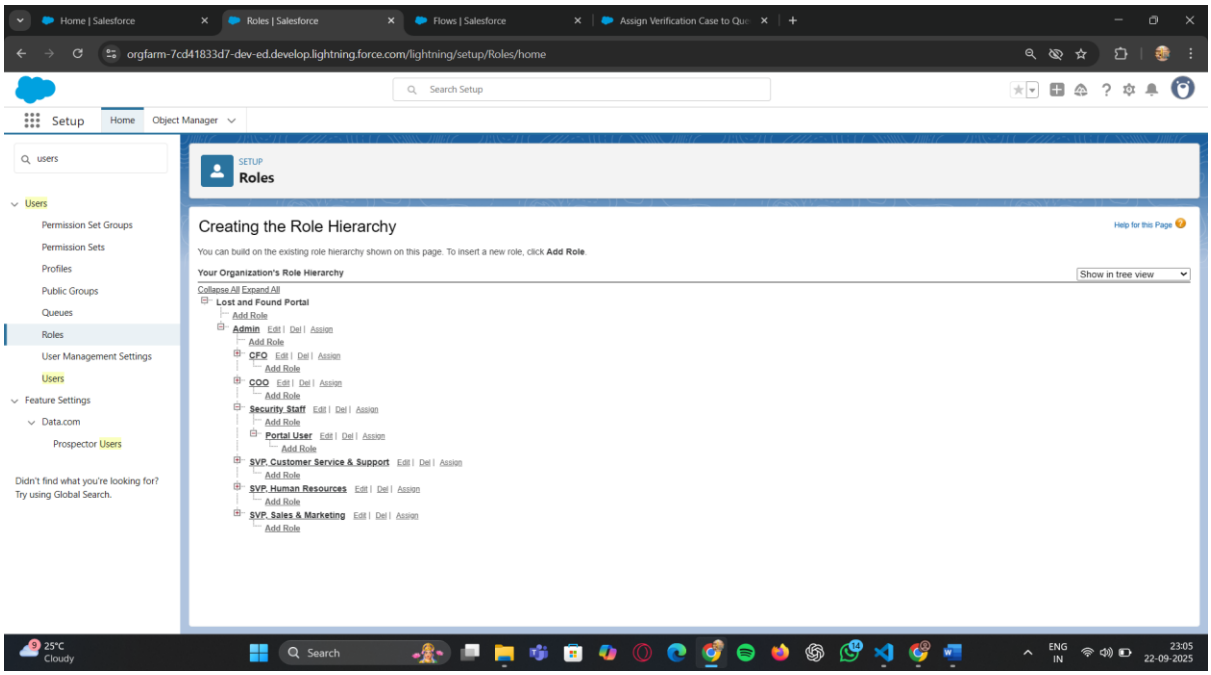


2.3 User & Security Model Setup

A security model was established to ensure users have appropriate access based on roles.

2.3.1 Roles

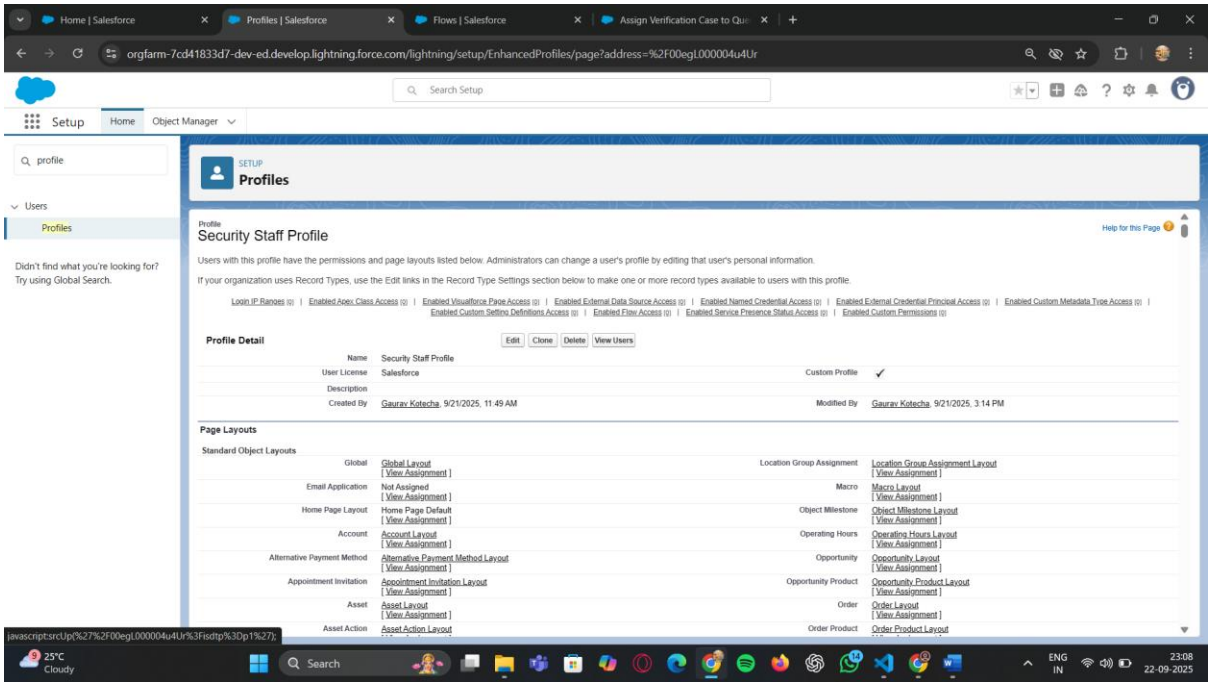
A role hierarchy was created to reflect the authority structure within the institution.



2.3.2 Profiles

A custom profile was created for the security team to grant them specific permissions to the Lost and Found objects.

- **Profile Name:** Security Staff Profile
- **Method:** Cloned from the standard "Standard User" profile.
- **Key Permissions:** Granted full Create, Read, Edit, and Delete (CRUD) access on the Lost Item, Found Item, and Verification Case objects.



2.4 Data Sharing & Visibility

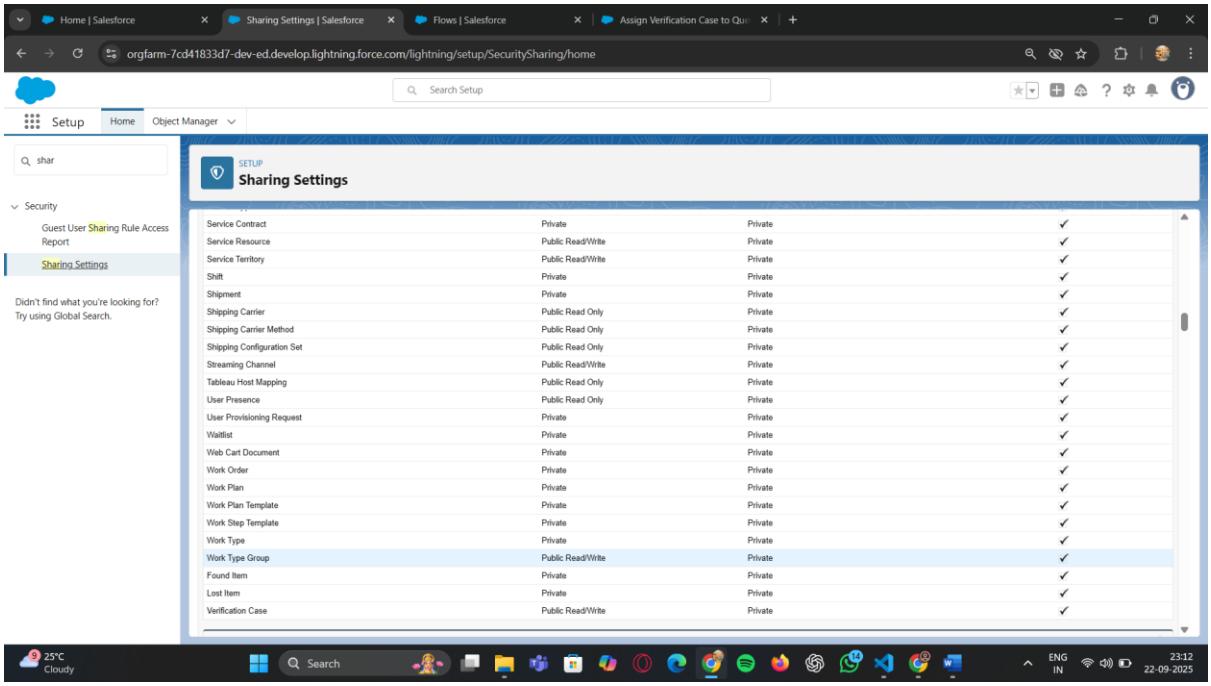
Default record visibility was set to private, with sharing rules created to grant access where neede

2.4.1 Organization-Wide Defaults (OWD)

The default internal access for the custom objects was set to the most restrictive level.

- **Lost Item:** Private
- **Found Item:** Private

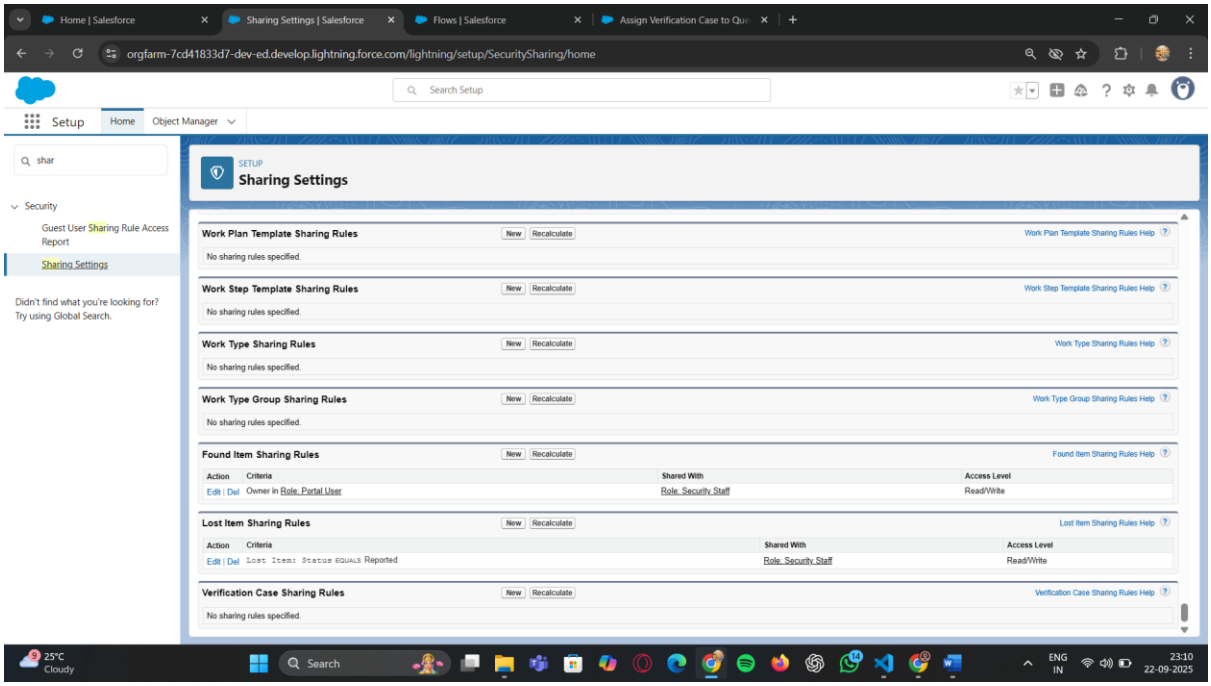
- **Verification Case:** Private



2.4.2 Sharing Rules

Rules were created to grant the Security Staff access to records they need to manage.

- **Rule 1:** Shared Lost Item records with the Security Staff role when the Status field equals "Reported".
- **Rule 2:** Shared Found Item records owned by Portal Users with the Security Staff role.


Home | Salesforce
Sharing Settings | Salesforce
Flows | Salesforce
Assign Verification Case to Qu
orgfarm-7cd41833d7-dev-ed.develop.lightning.force.com/lightning/setup/SecuritySharing/home
Search Setup
Setup
Home
Object Manager
shar
Security
Guest User Sharing Rule Access Report
Sharing Settings
Didn't find what you're looking for? Try using Global Search.
SETUP
Sharing Settings
Work Plan Template Sharing Rules
New
Recalculate
Work Plan Template Sharing Rules Help
No sharing rules specified.
Work Step Template Sharing Rules
New
Recalculate
Work Step Template Sharing Rules Help
No sharing rules specified.
Work Type Sharing Rules
New
Recalculate
Work Type Sharing Rules Help
No sharing rules specified.
Work Type Group Sharing Rules
New
Recalculate
Work Type Group Sharing Rules Help
No sharing rules specified.
Found Item Sharing Rules
New
Recalculate
Found Item Sharing Rules Help
Action
Criteria
Shared With
Access Level
Edit | Del
Owner in Role: Portal User
Role: Security Staff
Read/Write
Lost Item Sharing Rules
New
Recalculate
Lost Item Sharing Rules Help
Action
Criteria
Shared With
Access Level
Edit | Del
Lost Item: Status EQUALS Reported
Role: Security Staff
Read/Write
Verification Case Sharing Rules
New
Recalculate
Verification Case Sharing Rules Help
No sharing rules specified.
25°C
Cloudy
Search
ENG
IN
23:10
22-09-2025

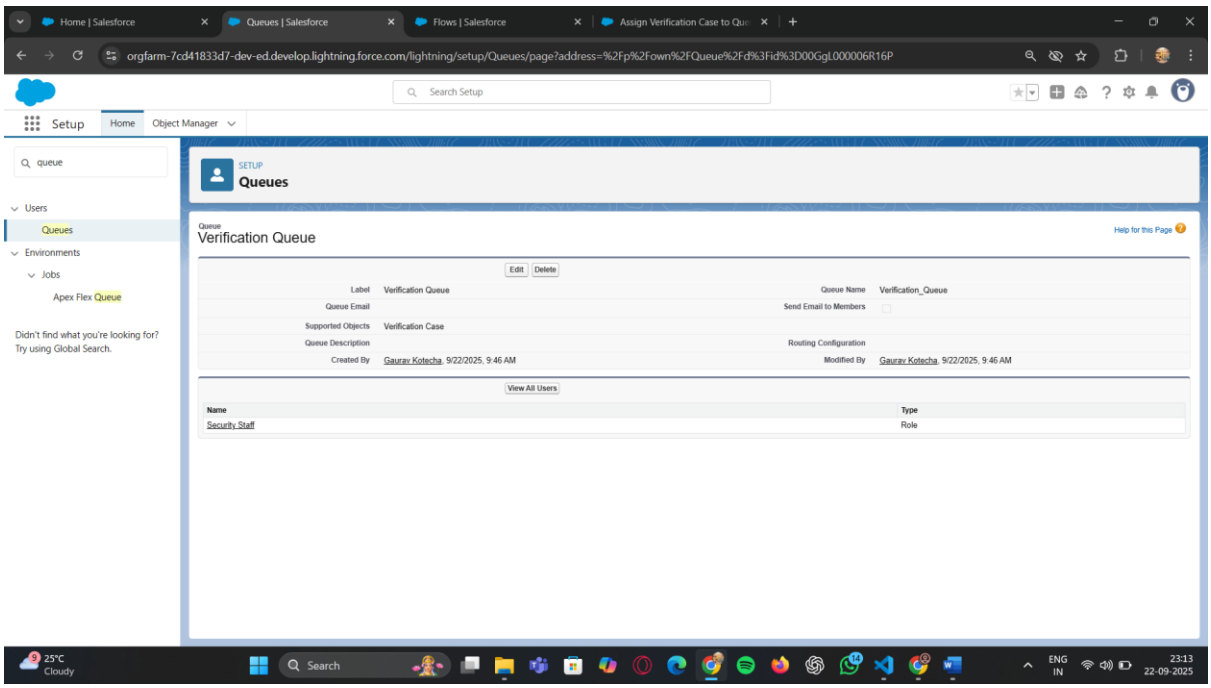
2.5 Automation Setup

Automation was built to streamline the case assignment process. Due to a platform bug, a workaround using Apex was required.

2.5.1 Queue

A queue was created to act as a central holding point for all new verification cases.

- **Queue Name:** Verification Queue
- **Supported Object:** Verification Case
- **Members:** Role: Security Staff



2.5.2 Invocable Apex (Workaround for Platform Bug)

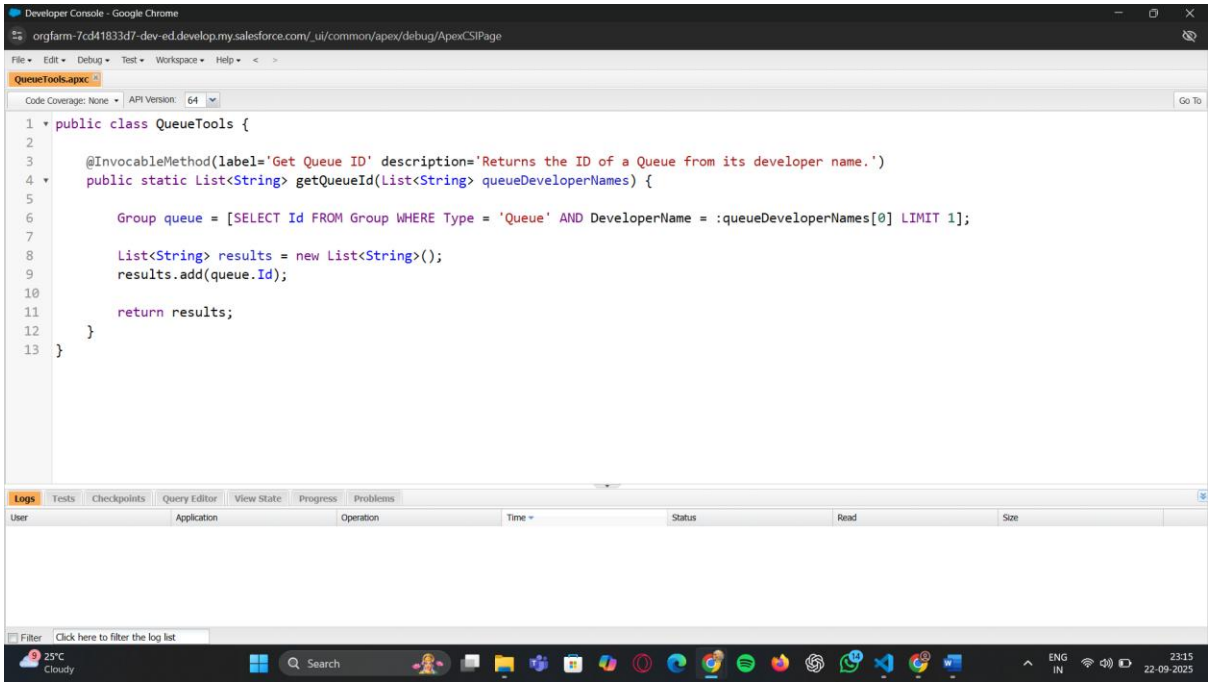
A bug in the Flow Builder prevented the direct selection of the Queue ID. To bypass this, a simple Apex class was created to find and return the Queue ID to the Flow.

- **Apex Class Name:** QueueTools
- **Code:**

Java

```
public class QueueTools {

    @InvocableMethod(label='Get Queue ID' description='Returns the ID of a Queue from its developer name.')

    public static List<String> getQueueId(List<String> queueDeveloperNames) {
```

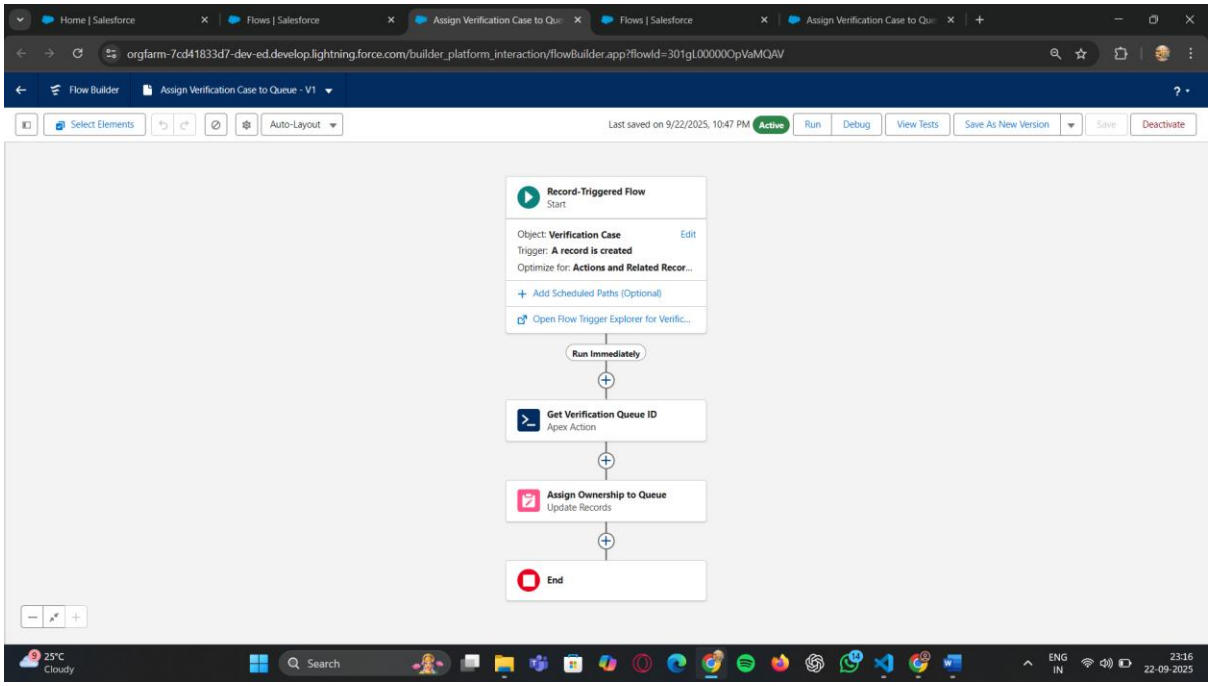
```
        Group queue = [SELECT Id FROM Group WHERE Type = 'Queue' AND
DeveloperName = :queueDeveloperNames[0] LIMIT 1];

        List<String> results = new List<String>();

        results.add(queue.Id);

        return results;

    }

}
```



2.5.3 Record-Triggered Flow

A Flow was built to automate the assignment of new Verification Cases.

- **Flow Name:** Assign Verification Case to Queue
- **Trigger:** Fires when a Verification Case record is created.
- **Logic:**
    1. **Action:** Calls the QueueTools Apex Action to get the Verification_Queue ID.
    2. **Update Records:** Sets the OwnerId of the new Verification Case record to the ID returned by the Apex Action.



2.6 Login Access Policies

Restricted Team Member login:

- Login Hours: 10 AM – 6 PM
- Managers have 24/7 access

2.7 Dev Org Setup

Verified that the Dev Org is ready for building automation (Flows, Validation Rules, etc.).

2.8 Sandbox Usage (Documentation Only)

For this project, Developer Org was used. In real deployments, work would be done in Sandbox, then deployed to Production via Change Sets or SFDX.

2.9 Deployment Basics

Used VS Code + Salesforce CLI (Ctrl+Shift+P) to deploy:

- Custom Objects
- Fields
- Profiles & Permission Sets