ServiceNow — Student Admission & Progress Management

Team ID: NM2025TMID18175

Team Members:

Team Leader: **MADESH.A**

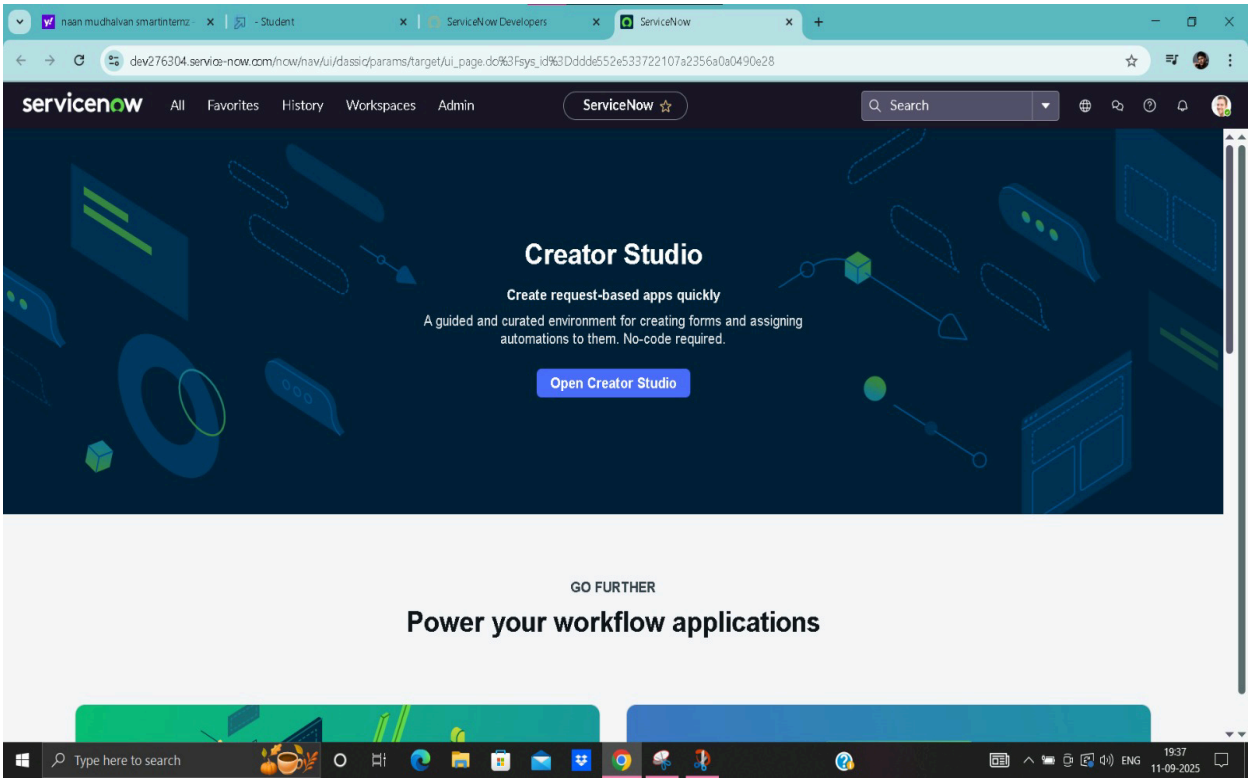
Member 1: **STERLIN RAJ.GS**

Member 2: **DEEPAK.N**

Member 3: **RAVINDRAN.S**

This document is a step-by-step project report created from project screenshots. It mirrors the structure and level of detail in your sample file. Each milestone contains activities and numbered steps that describe what to do in the ServiceNow UI.

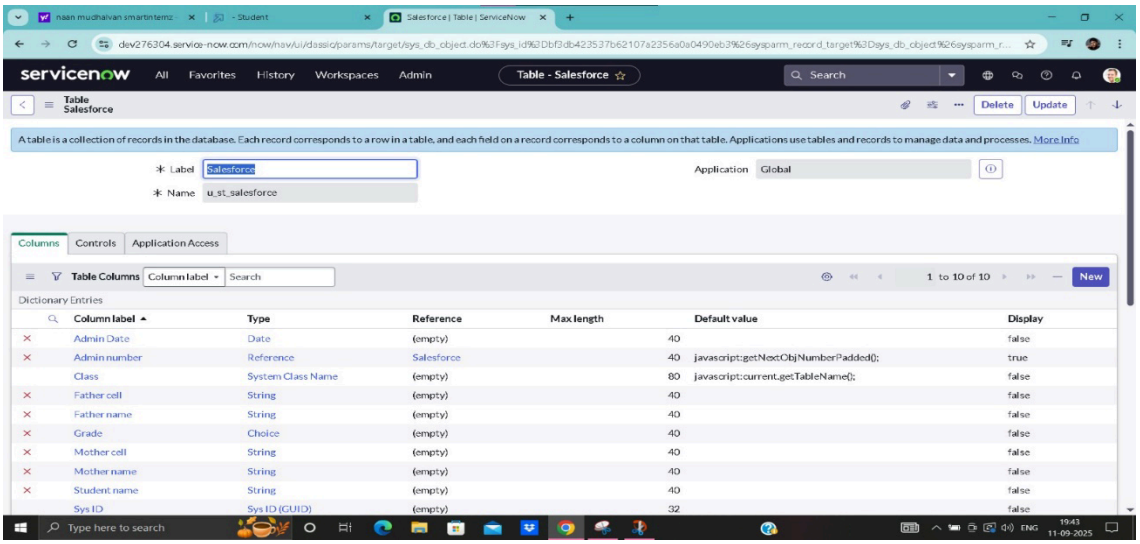
Milestone 1: Table Creation



Activity 1.1 — Create 'Salesforce' table

Purpose: store student personal and admission details.

1. Open ServiceNow and sign in to your developer instance.
2. Use the Application Navigator: click **All** and search for **Tables**.
3. Select **Tables** under **System Definition**.
4. Click **New** to create a table.
5. Fill the table form as follows:
6. - **Label:** Salesforce
7. - **Name:** u_st_salesforce (auto-generated but verify)
8. - **Application:** Global (or your scoped app)
9. Click **Submit** to save the new table.
10. Open the new table record and go to the **Columns** tab to add dictionary entries (fields).



Recommended columns (example):

• Admin Date — Type: Date

• Admin number — Type: Reference (self-reference for numbering or custom number field)

• Class — Type: String or System Class Name

• Father name — Type: String

• Father cell — Type: String

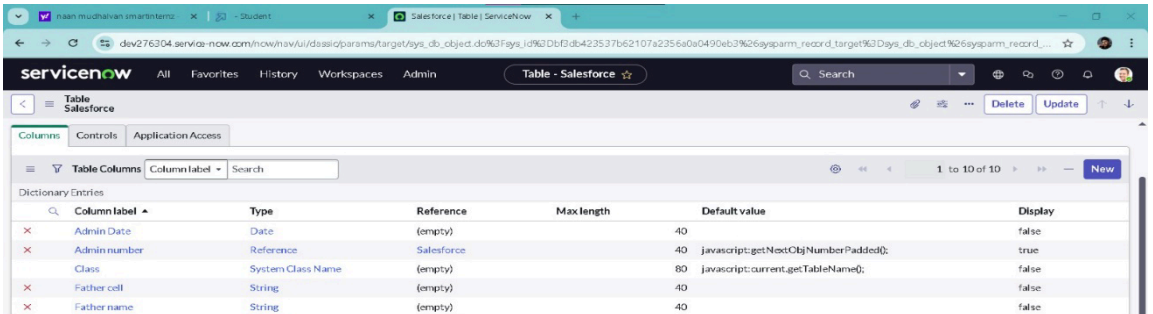
• Mother name — Type: String

• Mother cell — Type: String

• Student name — Type: String

• Grade — Type: Choice (create a choice list I..X)

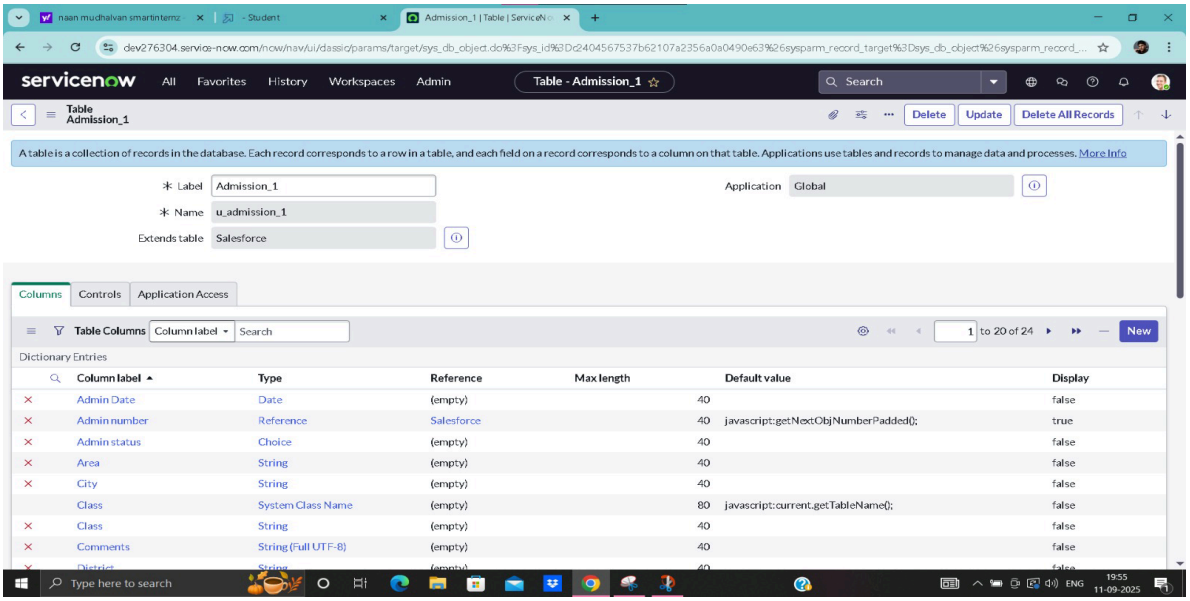
• Fee — Type: Price



Activity 1.2 — Create 'Admission_1' table (extends Salesforce)

Purpose: hold admission-specific records and extend the Salesforce table so shared fields are inherited.

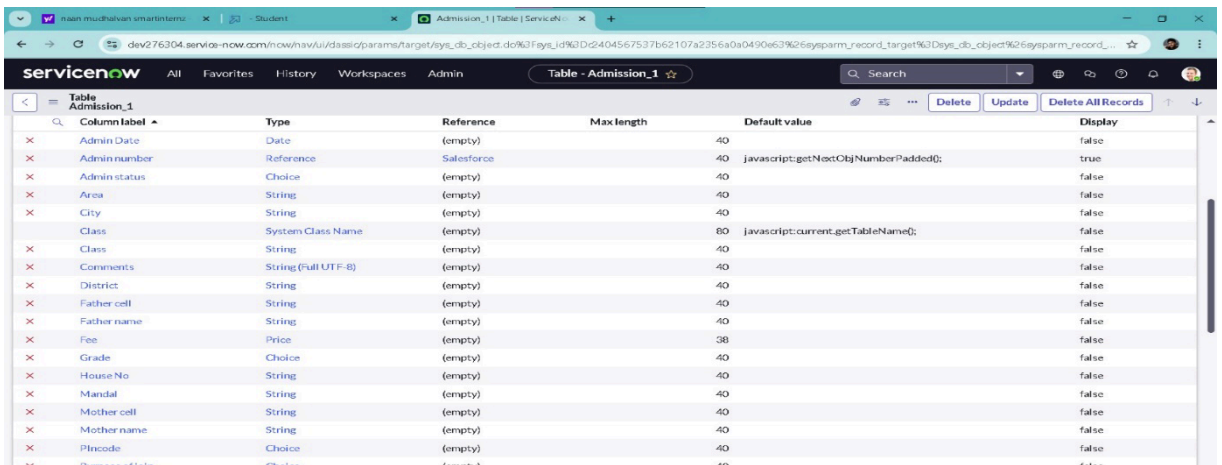
1. 1. Open **Tables** (All > Tables) and click **New**.
2. 2. Set **Label** to Admission_1 and **Name** to u_admission_1.
3. 3. In **Extends table**, select **Salesforce** (so Admission_1 inherits Salesforce fields).
4. 4. Check **Create module** & **Create mobile module** if you want a module in the application navigator (optional).
5. 5. Add/additional fields unique to Admission_1 (e.g., Address fields: Pincode, Area, City, District, House No).
6. 6. Click **Submit** to save.



Activity 1.3 — Create 'Students Progress' table

Purpose: capture marks and results for each student.

1. Create a new table **Students Progress** (u_students_progress) via All > Tables > New.
2. Add the following columns:
3. - Admission Number — Type: Reference -> Admission_1 (link to admission record)
4. - Telugu, English, Maths, Science, Social, Hindi — Type: Integer/String (marks)



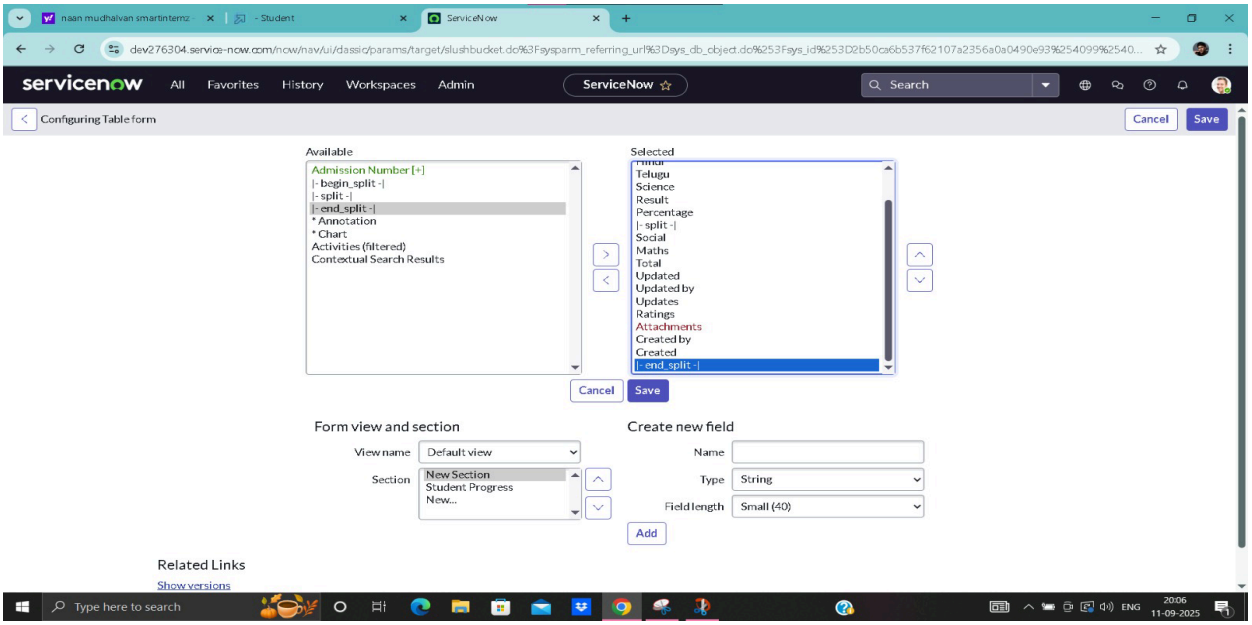
5. - Total — Type: Integer (auto-calculated)
6. - Percentage — Type: Decimal (auto-calculated)
7. - Result — Type: String/Choice (Pass/Fail or text)
8. Click **Submit** to save the table.

Milestone 2: Form Design & Layout

Activity 2.1 — Design Form for 'Salesforce' / 'Admission_1' / 'Students Progress'

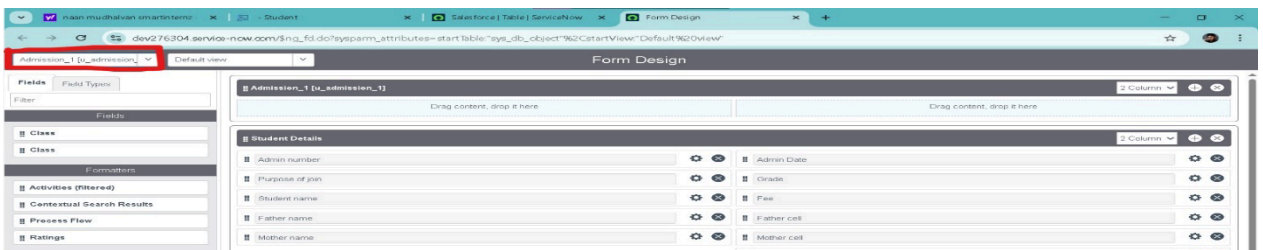
Use the Form Designer to create a user-friendly layout with sections and two-column rows.

7. 1. Open the table record (Salesforce, Admission_1, or Students Progress) from the Application Navigator (All > Tables > open the table).
8. 2. Click the **Form Design** link (usually a related link on the table record).
9. 3. In the Form Designer, add sections (for example: Student Details, School Details, Address, Student Progress).
10. 4. Drag and drop fields into the sections. Use a two-column layout for paired fields (e.g., Father name / Father cell).
11. 5. Use the gear icon to set field properties (mandatory, read-only, etc.) where needed.
12. 6. Click **Save** or **Update** when finished.



Activity 2.2 — Configure Form View and Related Links

13. 1. If necessary, create a custom view (Default view is OK for most cases).
14. 2. Use **Form Layout** or **Form Designer** to manage which fields appear on the main form vs related tabs.

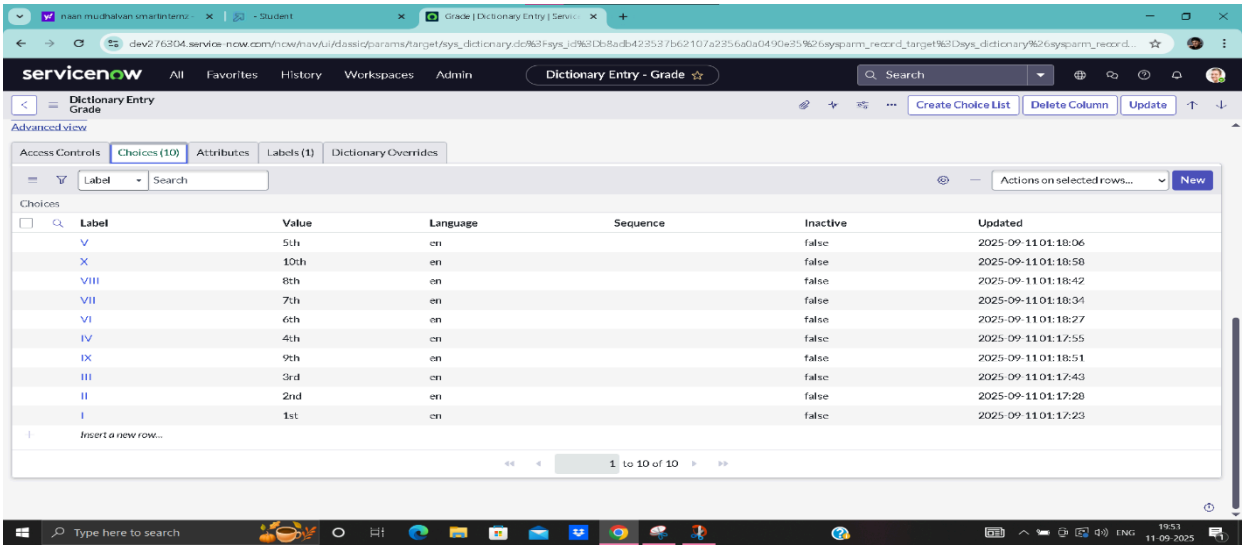


15. 3. Add the **Comments** field (or any Journal fields) in a single-column section for full-width input.
16. 4. Publish the changes and test by creating a new record (All > Salesforce_1 > New).

Milestone 3: Choice Lists / Dictionary Entries

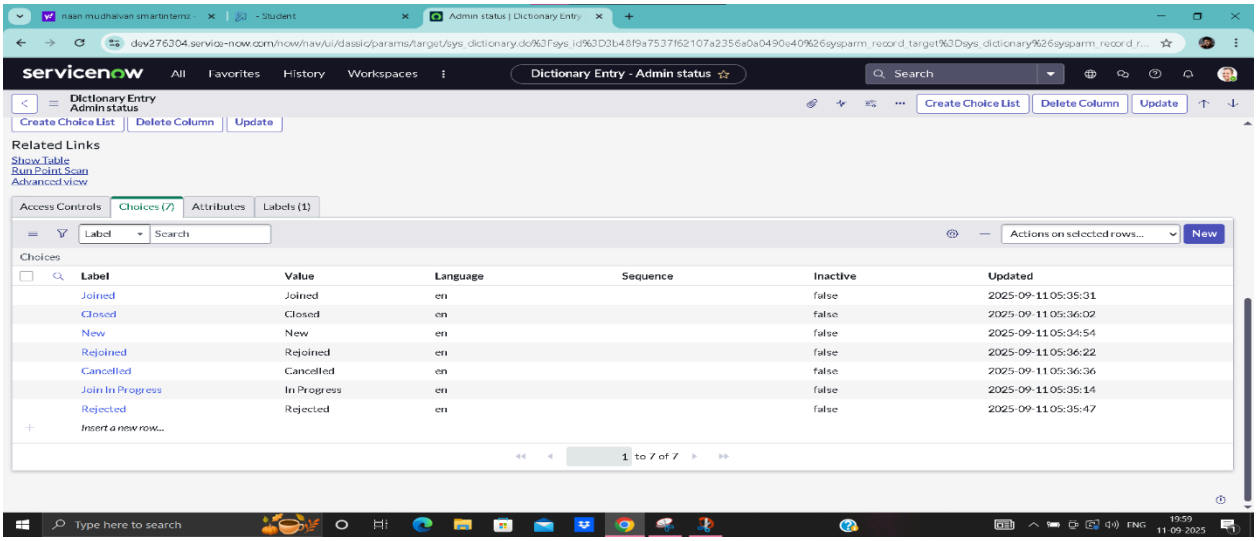
Activity 3.1 — Create 'Grade' choice list

17. 1. Open the **Dictionary** entry for the Grade column (All > System Definition > Dictionary and search Grade for the table).
18. 2. Click the **Choices** tab and add entries for the grade values (I, II, III, ..., X).
19. 3. Set the sequence/order so the list appears in the desired order in the form.
20. 4. Click **Update** to save.



Activity 3.2 — Create 'Admin status' and 'Purpose of join' choices

21. 1. Open the Dictionary entry for the Admin status column and add choices: New, Joined, Rejoined, Cancelled, Rejected, In Progress.



22. 2. For Purpose of join (field in Admission_1), create choices: Coaching, Tuition, Teacher.
23. 3. Update and test the dropdown choices in the form.

Milestone 4: Client Scripts (UI behavior & calculations)

Client scripts achieve immediate, client-side behavior without server calls. Below are the primary scripts implemented.

Activity 4.1 — Disable calculated fields onLoad

Goal: prevent users from editing fields that are auto-calculated (Total, Percentage, Result).

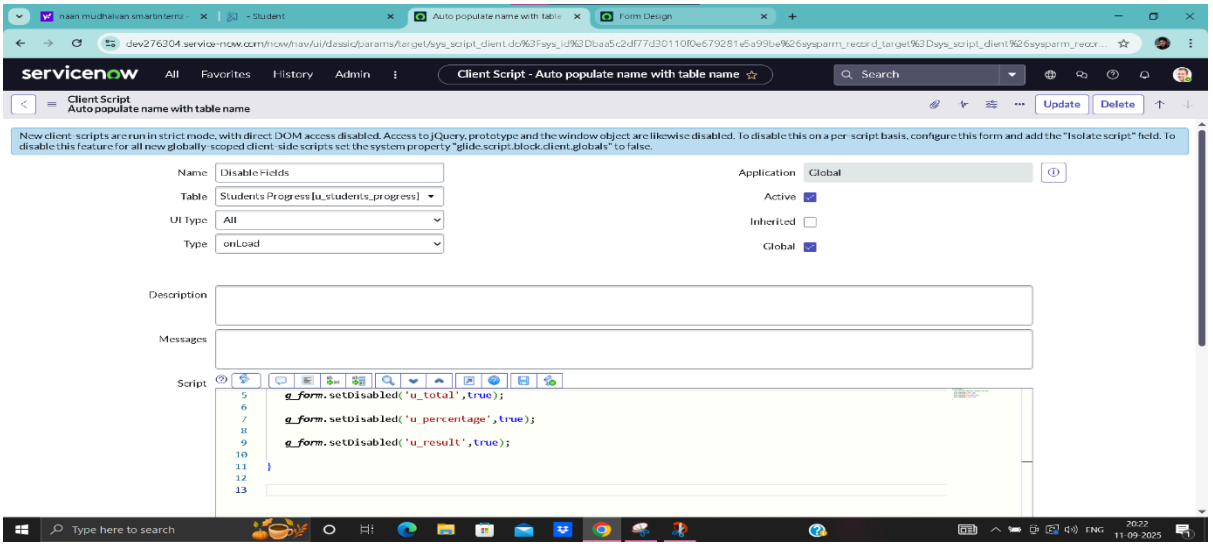
24. 1. Navigate to All > System UI > Client Scripts > New.
25. 2. Fill fields: Name = Disable Fields, Table = Students Progress [u_students_progress], UI Type = All, Type = onLoad.
26. 3. Script body example:

Example script (onLoad):

g_form.setDisabled('u_total', true);

g_form.setDisabled('u_percentage', true);

g_form.setDisabled('u_result', true);



Activity 4.2 — Calculate Total & Percentage (onChange)

Goal: when any subject mark changes, update the Total and Percentage fields automatically.

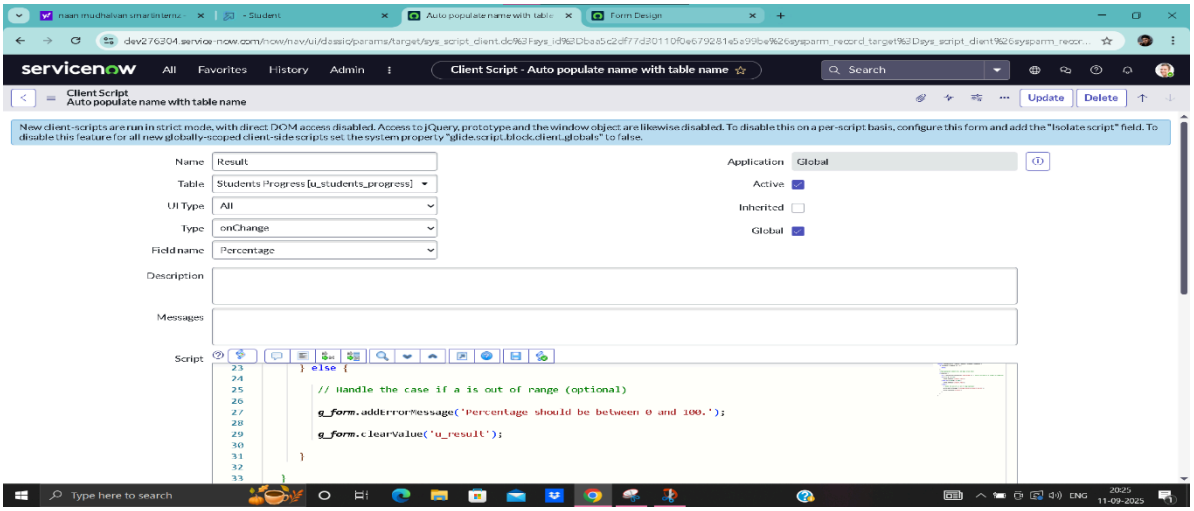
27. 1. Create a new client script for the Students Progress table (All > System UI > Client Scripts > New).
28. 2. Name = Total update, Table = Students Progress [u_students_progress], Type = onChange, Field name = (attach to each subject field OR create multiple scripts).
29. 3. Script should read each subject value (parseInt or treat NaN as 0), sum them, then set the Total and calculate Percentage.

30. 4. Click Update and test by changing marks on the form.

Example script:

```
function onChange(control, oldValue, newValue, isLoading) {
 if (isLoading) return;
 var a = parseInt(g_form.getValue('u_telugu')) || 0;
 var b = parseInt(g_form.getValue('u_english')) || 0;
 var c = parseInt(g_form.getValue('u_maths')) || 0;
 var d = parseInt(g_form.getValue('u_science')) || 0;
 var e = parseInt(g_form.getValue('u_hindi')) || 0;
 var f = parseInt(g_form.getValue('u_social')) || 0;
 var total = a + b + c + d + e + f;
 g_form.setValue('u_total', total);
 var percentage = (total / 600) * 100; // adjust divisor if different maximum per subject
 g_form.setValue('u_percentage', percentage.toFixed(2));
}
```

Activity 4.3 — Percentage validation & Result update



31. 1. Create a client script named Percentage (Table = Students Progress, Type = onChange, Field name = Total or Percentage).
32. 2. In the script, validate that percentage is between 0 and 100; if not, display an error and clear the Result field.
33. 3. Update the Result field based on percentage thresholds (e.g., >= 35 -> Pass, else Fail).

Example snippet:

```
if (percentage < 0 || percentage > 100) { g_form.addErrorMessage('Percentage should be between 0 and 100.'); g_form.clearValue('u_result'); }
```

Activity 4.4 — Auto-populate address from Pincode (onChange)

34. 1. Create a client script for Admission_1 (Name = Pincode update, Table = Admission_1, Type = onChange, Field name = Pincode).
35. 2. Implement a simple lookup (hardcoded mappings or use GlideAjax / REST to a postal API) to populate Area, City, District.
36. 3. Example simple script (hardcoded mapping):

Example mapping code:

```
var pin = g_form.getValue('u_pincode');
if (pin === '500081') {
 g_form.setValue('u_area', 'Pattabiram');
 g_form.setValue('u_city', 'Chennai');
 g_form.setValue('u_district', 'Tiruvallur');
}
```

Activity 4.5 — Auto-populate fields from referenced Admission number

37. 1. Create a client script on Admission Number (onChange) to pull details from the referenced Admission record using g_form.getReference().
38. 2. Use the callback to set fields: g_form.setValue('u_father_name', ref.u_father_name) etc.
39. 3. Test by selecting an Admission Number reference value and ensuring fields populate.

Example pattern (client-side):

```
g_form.getReference('u_admission_number', function(ref) {
 if (ref) {
  g_form.setValue('u_father_name', ref.u_father_name);
  g_form.setValue('u_mother_name', ref.u_mother_name);
  g_form.setValue('u_father_cell', ref.u_father_cell);
 }
});
```

Milestone 5: Automatic Numbering (System Numbers)

Activity 5.1 — Create Number record

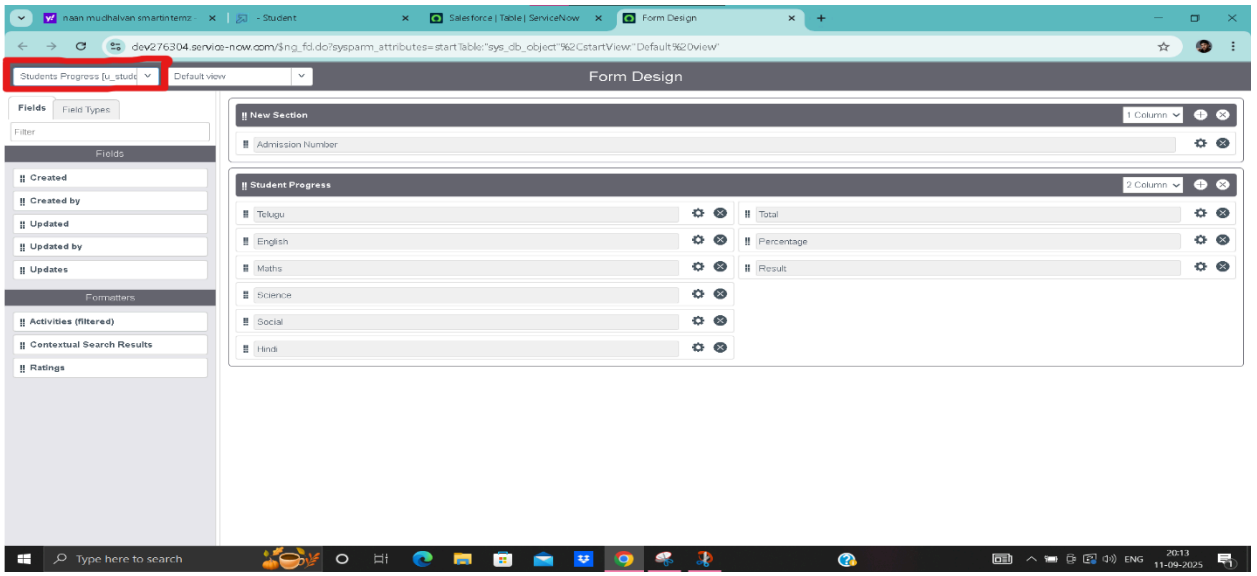
40. 1. Navigate to All > System Definition > Numbers (or Number Maintenance).
41. 2. Click New to create a numbering rule.
42. 3. Set Table = Salesforce (or the table to auto-number), Prefix = SAL, Number = starting value (e.g., 1000), Number of digits = 7.
43. 4. Click Update to save. New records created will use the SAL prefix with zero-padded digits.

Milestone 6: Flow Formatter & Flow Designer

Activity 6.1 — Create Flow Formatter for Admission_1

44. 1. All > Flow Formatter > New.
45. 2. Set Table = Admission_1, Name/Label = New (or appropriate label).

46. 3. Add a Condition, for example: Admin status is New.
47. 4. Save — the flow formatter will display the label/formatting on list and form based on conditions.



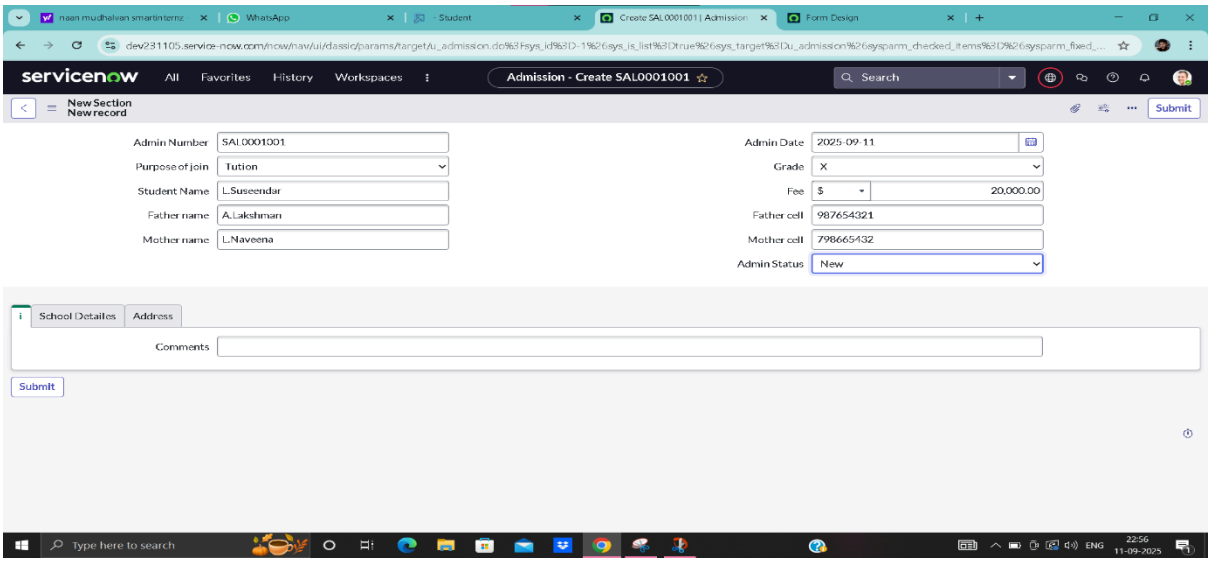
Activity 6.2 — Create a Flow in Flow Designer (optional)

48. 1. Open Flow Designer (All > Flow Designer).
49. 2. Create a new Flow and choose trigger (e.g., create record on Students Progress).
50. 3. Add actions: Update Record (set status to Completed), Ask for Approval (assign to approver), etc.
51. 4. Activate the Flow and test by creating a record that meets the trigger condition.

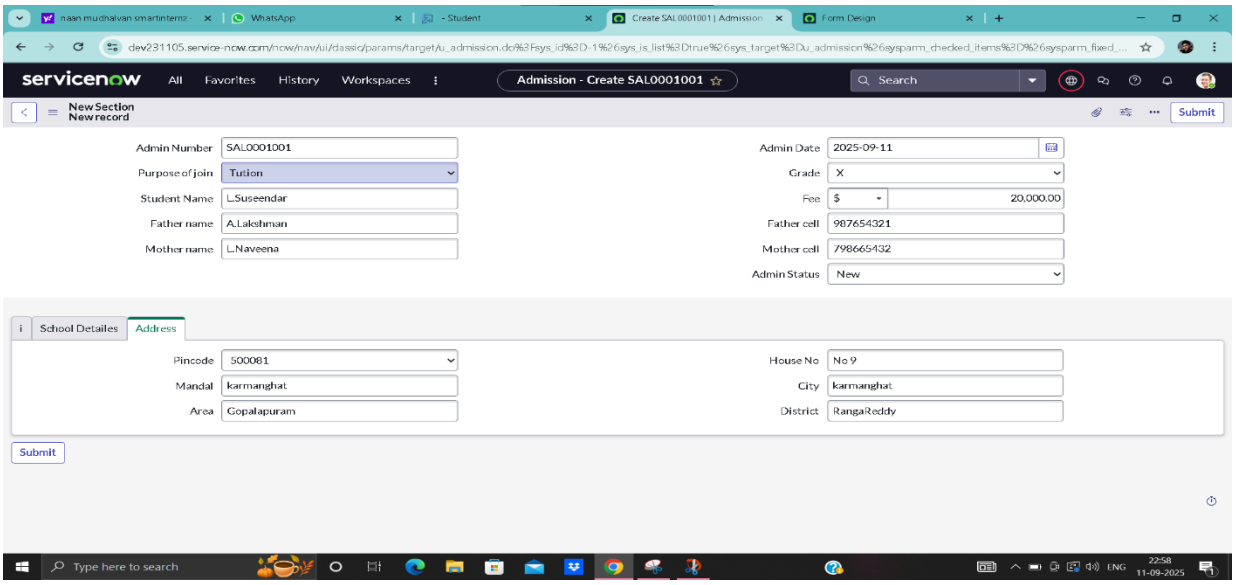
Milestone 7: Data Entry, Validation & Testing

Activity 7.1 sample Admission & Salesforce records

52. 1. Use the application module (All > your app > Salesforce_1 or Admission_1) and click New.

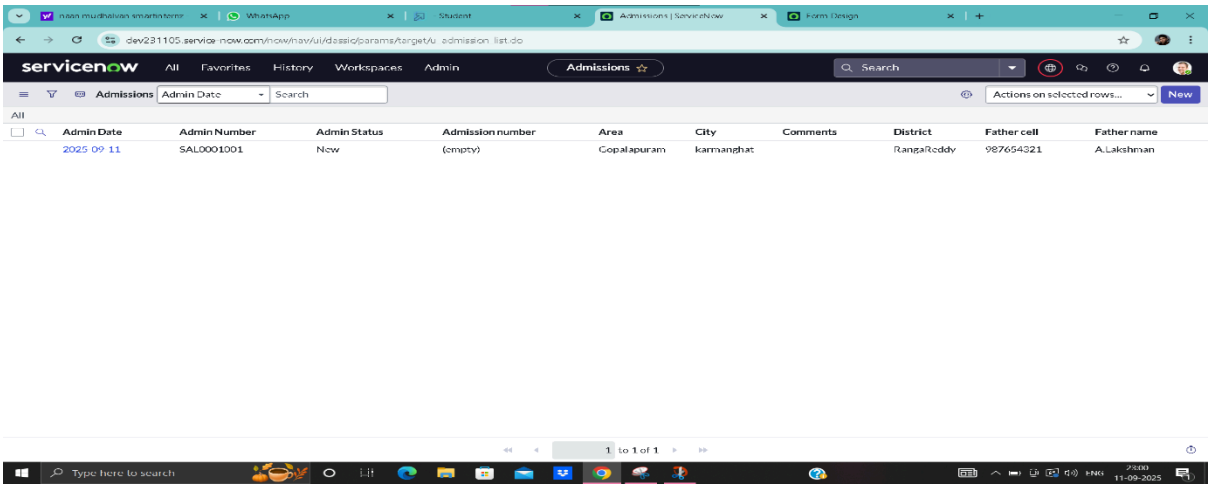


53. 2. Fill sample records (Admin number, Student name, Parents info, Grade, Fee, Address) and Submit.
54. 3. Verify the auto-numbering (SAL...), and that referenced fields populate into Students Progress when selected.
55. 4. Create Students Progress records and verify Total, Percentage, and Result are computed automatically.



Activity 7.2 — List view and data verification

56. 1. Open the list view for Salesforce_1 table to confirm records display properly (Admin Date, Admin number, Parent names, Student name, Grade).
57. 2. Use filter and search boxes on the list headers to validate searching and sorting behavior.



Conclusion

In this project, We succesfully developed an educational organisation management system using servicenow. This project demonstrates how ServiceNow can be applied beyond IT services,providing useful solutions in an educational environment.