Educational Organisation Using ServiceNow Project Report

Team Details

Faculty Mentor: Dr Touseef Ahamad
Team Leader: Nageti Kuladeep

---

1. INTRODUCTION

1.1 Project Overview

This project involves developing a **ServiceNow-based application** to manage key academic workflows within an educational organization. The project focuses on automating the student admission process and tracking academic performance through well-structured forms, custom tables, and client-side scripting.

1.2 Purpose

To replace manual, repetitive educational processes with a centralized, automated ServiceNow solution that enhances data accuracy, improves efficiency, and streamlines academic record handling using minimal code.

---

2. Ideation Phase

2.1 Problem Statement

Educational institutions face significant delays and inaccuracies due to manual admissions and progress tracking. These inefficient systems lack automation, real-time updates, and structured form validations.

2.2 Empathy Map Canvas

- **Users** : Admission Officers, IT Admins, Students
- **Needs** : Fast form filling, automatic calculations, reliable data
- **Pain Points** : Manual entry errors, duplicated work, unclear workflows

2.3 Brainstorming

- Design three main tables for Admissions, Salesforce, and Progress
- Create custom forms using Form Designer
- Use client scripts for field calculations, validations, and data population
- Automate workflows with Flow Designer

---

3. Requirement Analysis

3.1 Customer Journey Map

1. Admission form is filled
2. Data flows into Admission table
3. Academic scores are entered into Progress table
4. Client scripts auto-calculate results
5. Admin verifies and stores student records

3.2 Solution Requirement

- Custom tables
- Custom number maintenance

- Form design
- Client scripts
- Flow automation
- UI policies (optional)

3.3 Technology Stack ●

Platform : ServiceNow ●

Tools Used :

- Table Designer
- Form Designer
- Flow Designer
- Update Sets
- Script Editor (Client Scripts)

- **Script Types** : onChange, onLoad

---

4. Project Design Phase

4.1 Problem-Solution Fit

The application automates data population, validation, and result generation. It reduces manual steps while improving form reliability.

4.2 Proposed Solution

-
-
  Use of 3 custom tables: Admissions, Salesforce, Progress

  Custom forms for user interaction

- Automated field behavior using client scripts
- Flow Designer to automate student entry validation process

4.3 Solution Architecture

Architecture Flow :
Form Entry → Table Record Creation → Client Script Execution → Auto Calculations/Disabling Fields → Output Storage

---

5. Project Planning Phase

5.1 Project Planning

Week	Task	Tools Used
1	Setup ServiceNow Instance	ServiceNow Personal Instance
2	Create Tables & Update Set	Table Designer
3	Form Layout and Number Maintenance	Form Designer
4	Write Client Scripts	Script Editor
5	Testing and Final Output Verification	Form UI, Script Logs

-
-

6. Performance Testing

6.1 Performance Testing

- **Form Load Time** : Optimized and responsive
- **Script Execution** : Fast and accurate

Field Calculations : Worked flawlessly on every input

Process Flow : Triggered as expected under different scenarios

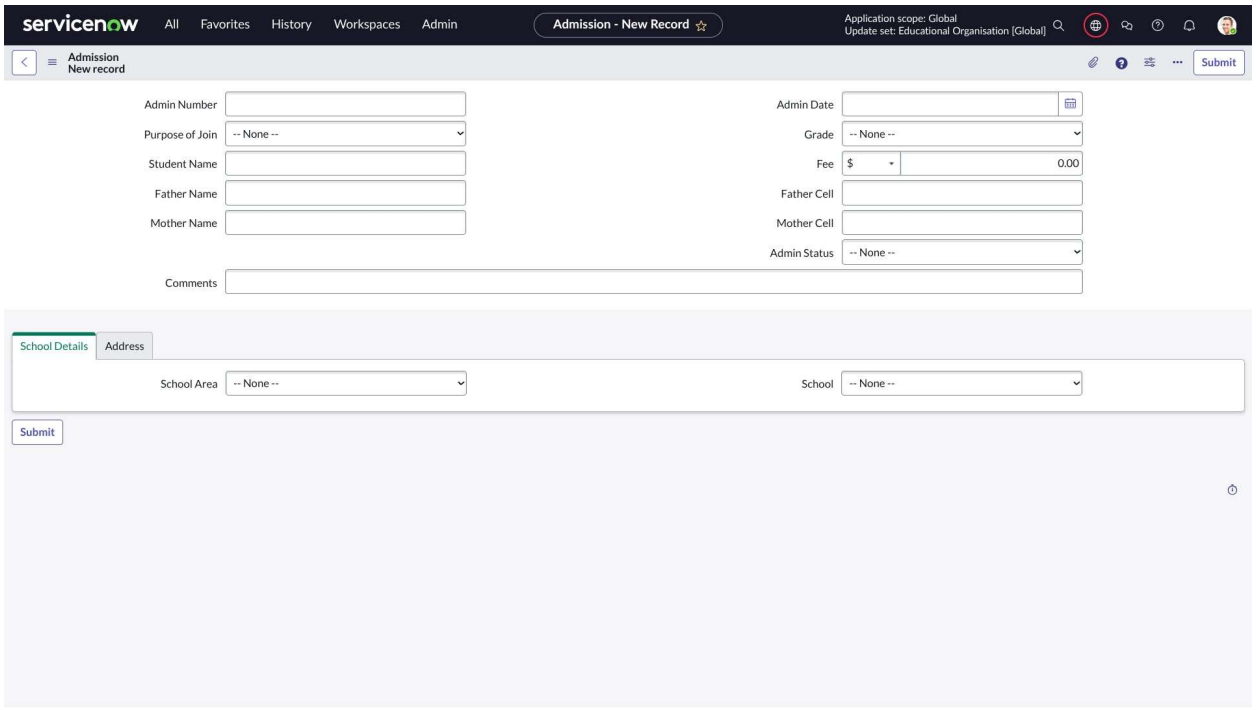
---

7. Results

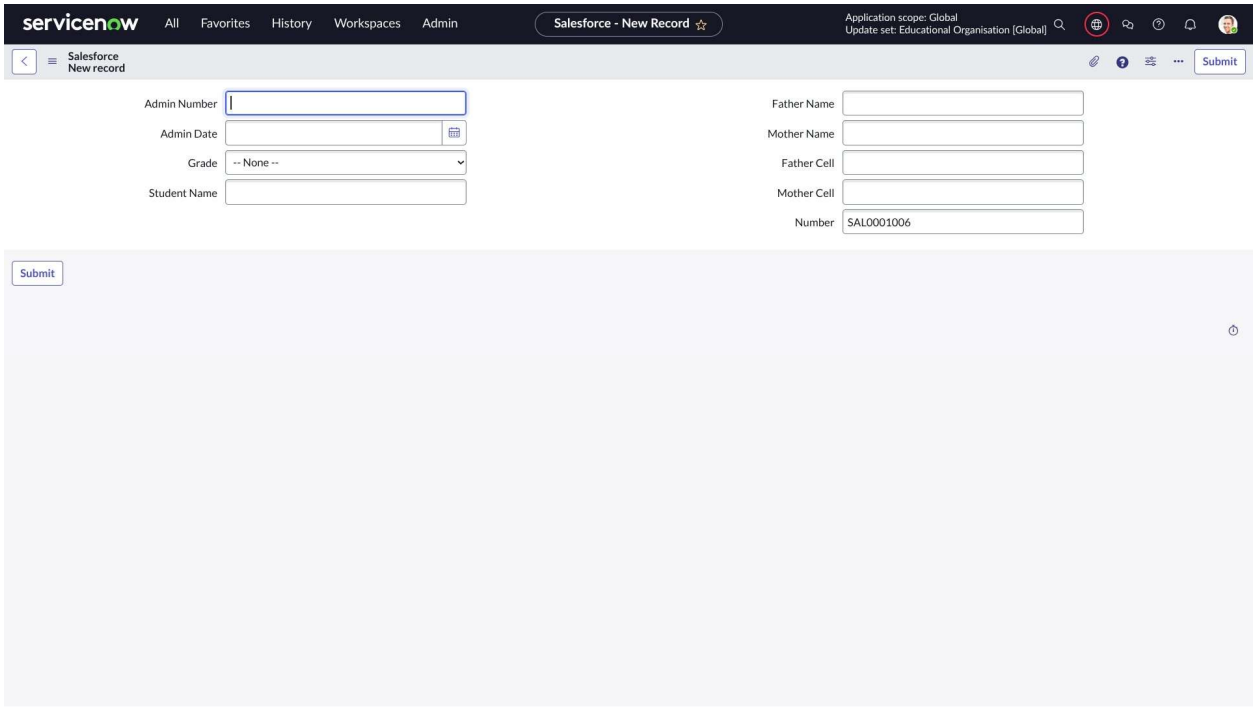
7.1 Output Screenshots

Screenshots

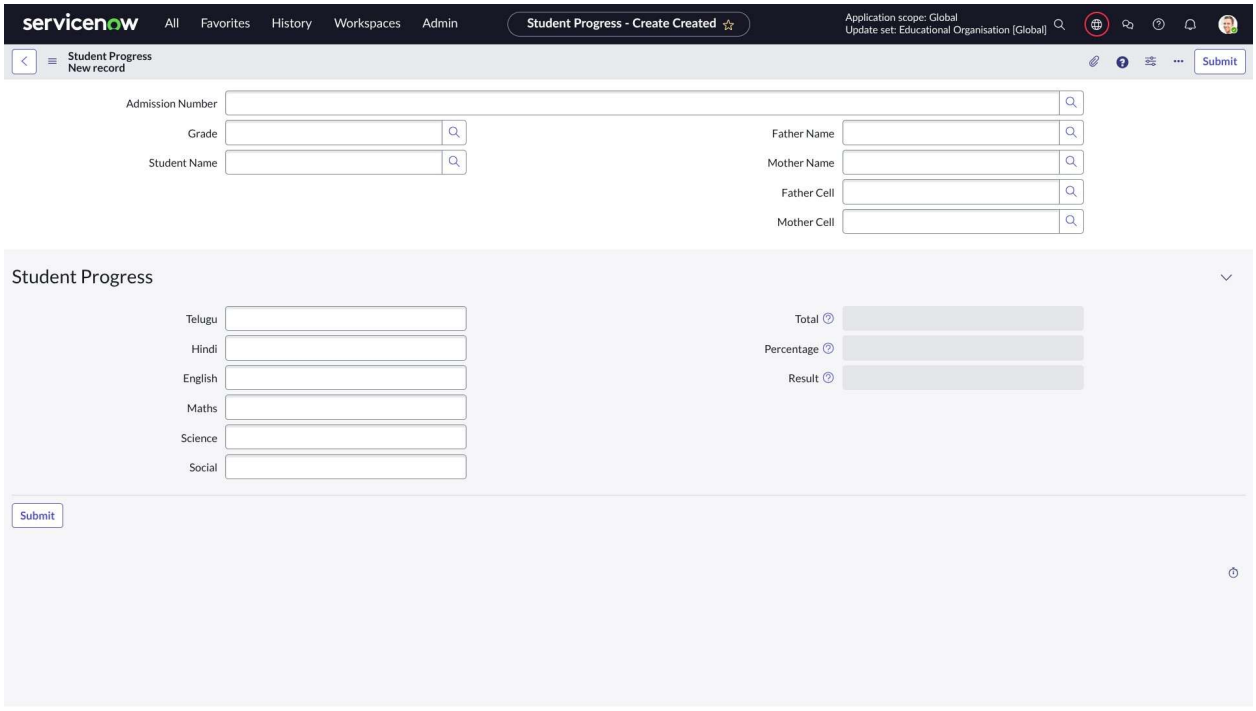
Admission Table Form:



Saleforce Table Form:



Student Progress Table Form:



8. Advantages & Disadvantages

Advantages

- Low-code development
- High scalability and maintainability
- Real-time calculations
- Structured, clean UI

Disadvantages

- Steep learning curve for new users
- Limited by ServiceNow's UI flexibility
- Complex logic may require JavaScript skills

---

9. Conclusion

This project successfully demonstrates the capability of ServiceNow to digitize and automate educational workflows. With minimal scripting and smart configuration, the solution ensures better accuracy, speed, and user experience.

---

10. Future Scope

- Add dashboards for analytics
- Enable role-based access controls
- Send automated notifications (email/SMS)
- Integration with external reporting tools

- Improve mobile accessibility via Service Portal

---

11. APPENDIX

Client Scripts used are:

1. Auto Populate (Admission Table – onChange)

```
function onChange (control, oldValue, new Value, isLoading,
    isTemplate)       { if ( isLoading ||   new Value ===   '' )
  return ;   var a   =  g_form.getReference( 'u_admission_number' );
  g_form.setValue( 'u_admin_date' ,  a.u_admin_date);
  g_form.setValue('u_grade'       ,   a.u_grade);  g_form.setValue(
  'u_student_name',  a.u_student_name);   g_form.setValue(
  'u_father_name' ,  a.u_father_name);  g_form.setValue(
  'u_mother_name',  a.u_mother_name);  g_form.setValue(
    'u_father_cell'   ,  a.u_father_cell);   g_form.setValue(
  'u_mother_cell' ,  a.u_mother_cell);  g_form.setDisabled(
  'u_admin_date',  a.u_admin_date);   g_form.setDisabled( 'u_grade',
  a.u_grade);   g_form.setDisabled( 'u_student_name',
  a.u_student_name); g_form.setDisabled(       'u_father_name',
  a.u_father_name);  g_form.setDisabled( 'u_mother_name' ,
  a.u_mother_name);  g_form.setDisabled( 'u_father_cell' ,
  a.u_father_cell);  g_form.setDisabled( 'u_mother_cell' ,
  a.u_mother_cell);
}
```

2. Pincode Update (Admission Table – onChange)

```
function onChange (control, oldValue, new Value , isLoading,
  isTemplate) { if ( isLoading || new Value === '' ) return ;
  var a = g_form.getValue( 'u_pincode' ); if (a == '509358' )
  { g_form.setValue( 'u_mandal' , 'kadthal' ); g_form.setValue(
  'u_city' , 'kadthal') ;
      g_form.setValue( 'u_district' , 'RangaReddy' );
   } else if (a == '500081' ) {
     g_form.setValue( 'u_mandal' ,
     'karmanghat' );
```

```
      g_form.setValue( 'u_city' , 'karmanghat' );
    g_form.setValue( 'u_district' , 'RangaReddy' );
  } else if (a == '500079' ) {
    g_form.setValue( 'u_mandal' , 'Abids' );
    g_form.setValue( 'u_city' , 'AsifNagar'
    ); g_form.setValue( 'u_district' ,
    'Hyderabad' );
  }
}
```

3. Disable Fields (Student Progress Table – onLoad)

```
function onLoad () { g_form.setDisabled( 'u_total',
  true ); g_form.setDisabled( 'u_percentage' , true
  ); g_form.setDisabled( 'u_result' , true );
}
```

4. Total Update (Student Progress Table – onChange)

```
function onChange (control, oldValue, newValue, isLoading,
  isTemplate) { if (isLoading || newValue === '') return ;
  if (newValue) { var a = parseInt (g_form.getValue( 'u_telugu'
  )); var b = parseInt (g_form.getValue( 'u_hindi' )); var c =
  parseInt (g_form.getValue( 'u_english' )); var d = parseInt
  (g_form.getValue( 'u_maths' )); var e = parseInt
  (g_form.getValue( 'u_science' )); var f = parseInt
  (g_form.getValue( 'u_social' )); var Total = parseInt (a + b
  + c + d + e + f); g_form.setValue( 'u_total' , Total);
   }
}
```

5 . Result Calculation (Student Progress Table – onChange)

```
function onChange (control, oldValue, new Value , isLoading,
  isTemplate) { if (isLoading || new Value === '' ) return ;
  if ( new Value ) {
```

```
      var a = parseInt(g_form.getValue(
     'u_percentage' )); if (a >= 0 && a <= 59
     ) { g_form.setValue( 'u_result' , 'Fail' );
      } else if (a >= 60 && a <= 100 ) {
        g_form.setValue( 'u_result' , 'Pass' );
      } else { g_form.addErrorMessage( 'Percentage should be
        between 0 and 100.' ); g_form.clearValue( 'u_result' );
      }
   }
}
```

6. Percentage Calculation (Student Progress Table – onChange)

```
function onChange (control, oldValue, new Value , isLoading,
  isTemplate) { if (isLoading || new Value === '' ) return ;
  var Total = g_form.getValue( 'u_total' ); var Percentage =
  (Total / 600 ) * 100 ;
g_form.setValue( 'u_percentage' , Percentage + '%' );
}
```