**Educational Organisation Using**

**ServiceNow Project Report**

# Team Details

**Faculty Mentor:**  Dr Touseef Ahamad

**Team Leader:**  Nageti Kuladeep

# INTRODUCTION

## 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.

## 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.

# Ideation Phase

## 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.

## 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

## 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

# Requirement Analysis

## 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

## 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

# Project Design Phase

## Problem-Solution Fit

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

## 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

## Solution Architecture

**Architecture Flow** :

Form Entry → Table Record Creation → Client Script Execution → Auto Calculations/Disabling

Fields → Output Storage

# Project Planning Phase

## 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 |

# Performance Testing

## Performance Testing

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

**Field Calculations** : Worked flawlessly on every input

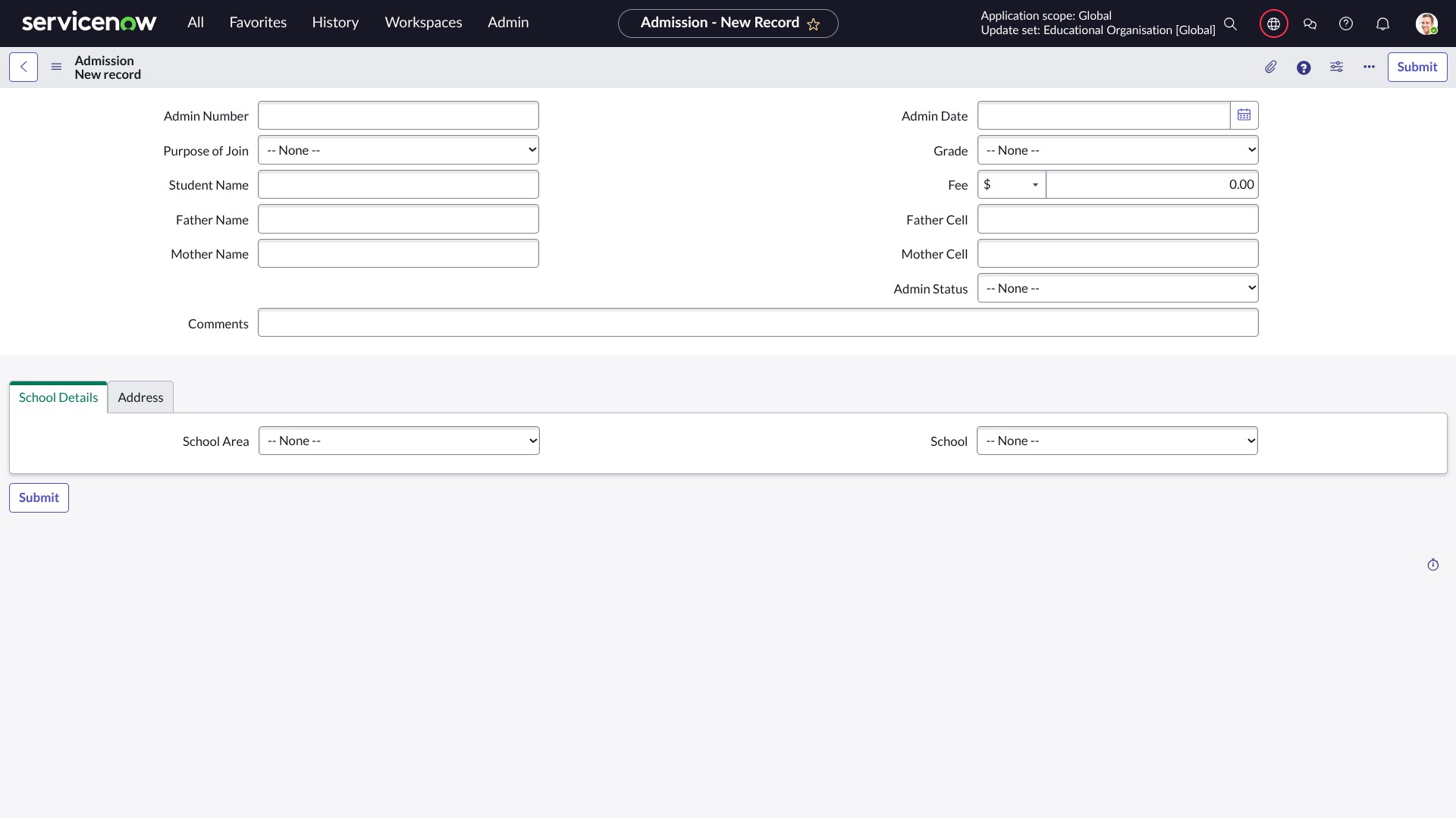
**Process Flow** : Triggered as expected under different scenarios

# Results

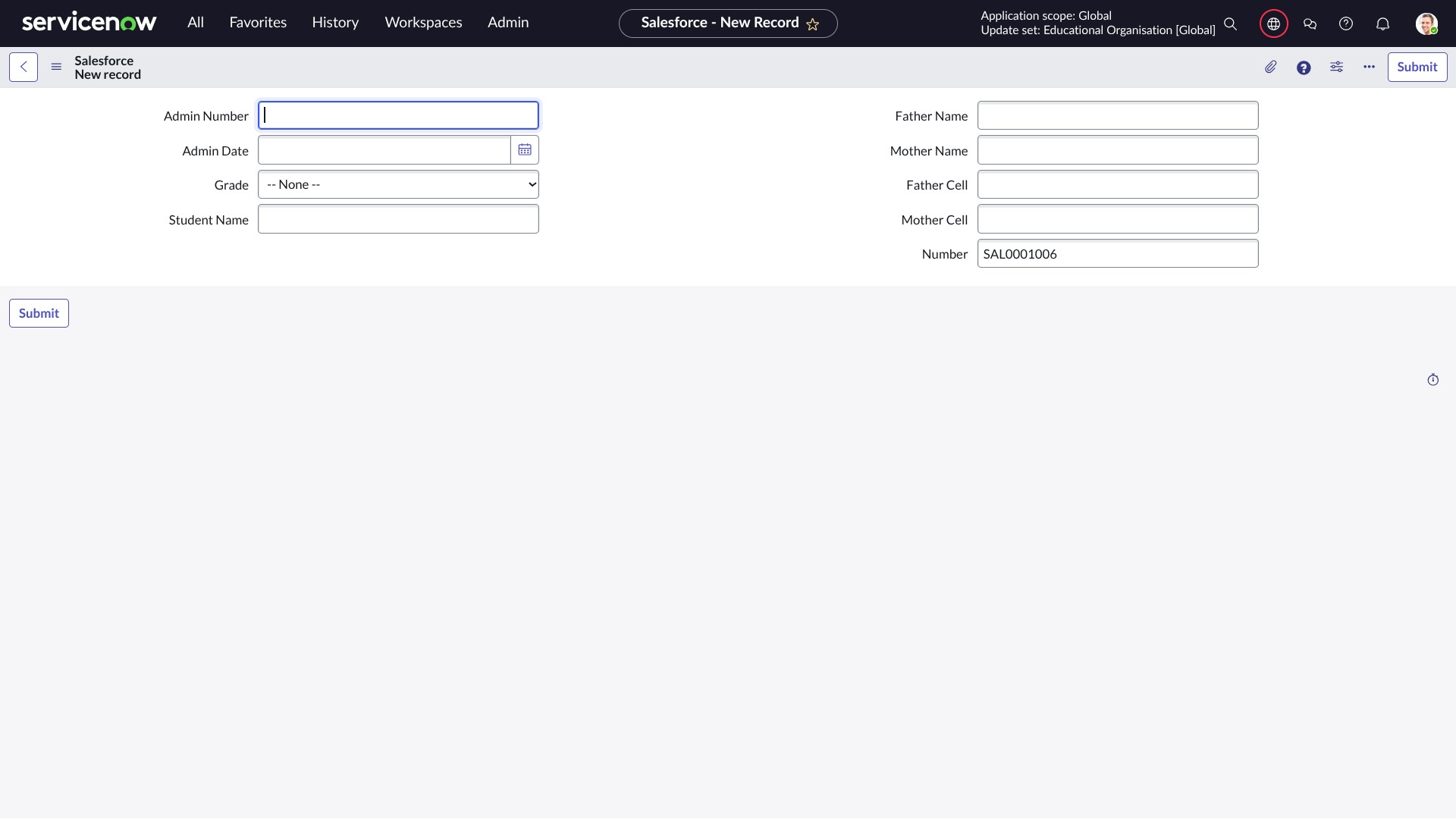
## Output Screenshots

Screenshots

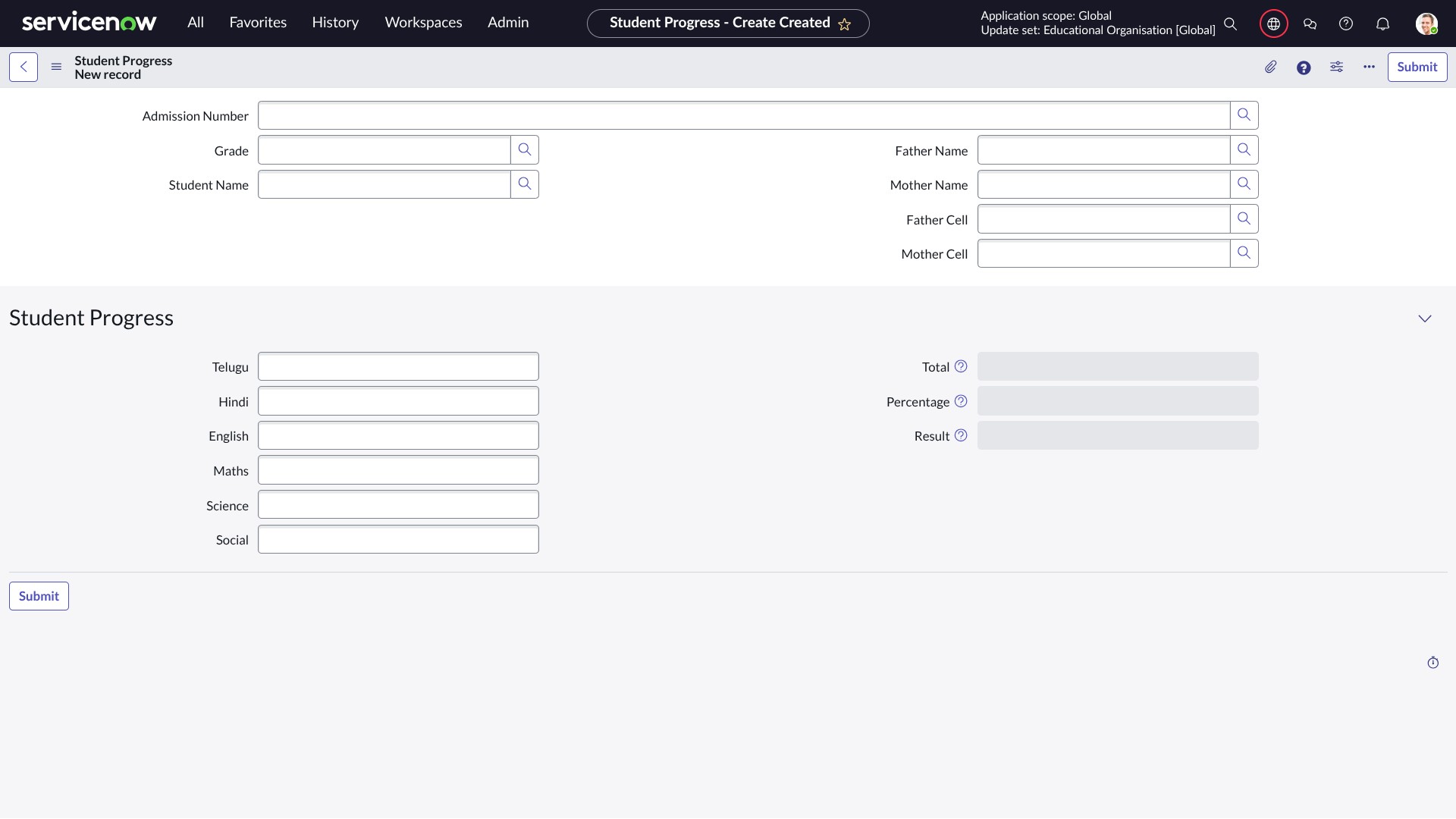
**Admission Table Form:**



**Saleforce Table Form:**



**Student Progress Table Form:**



# 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

# 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.

# 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

# 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);

}

1. **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' );  }  } |

1. **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 );

}

1. **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 + '%' );

}