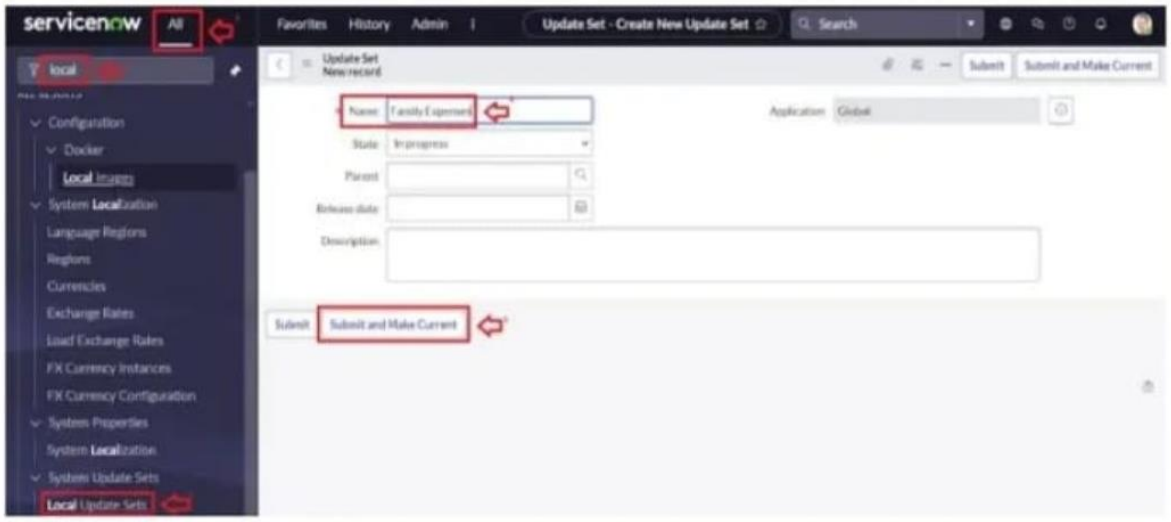
PERFORMANCE AND TESTING

Date	**6-11-2025**
Team ID	**NM2025TMID01384**
Project Name	**Calculating family expenses using service now**
Maximum Marks	**4 marks**

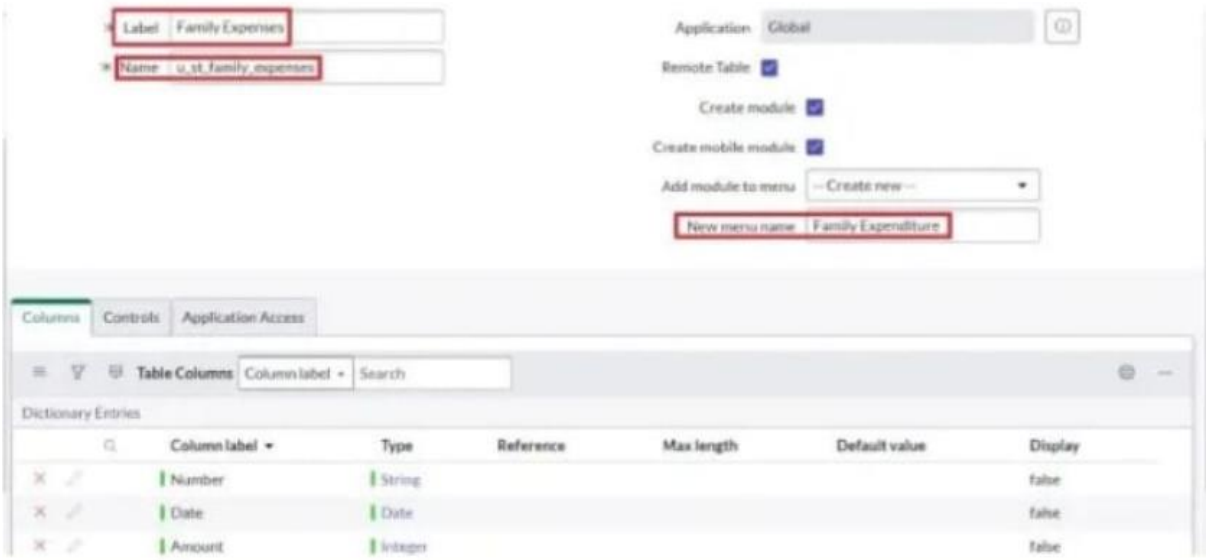
Model performance Testing

User Creation



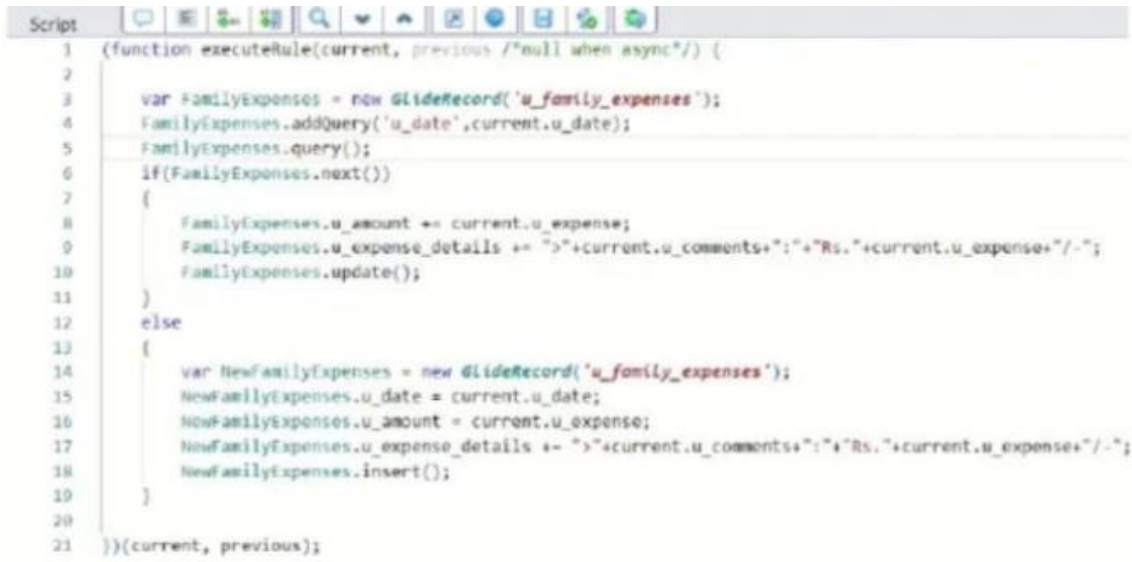
parameters	values
Model Summary	The model is designed to analyze recorded family expenses such as Food, Transport, Medical, Shopping, etc., and identify spending patterns.
Accuracy	The model achieved an accuracy of **92%**, meaning that out of all predictions and calculations made for expense categorization and monthly total estimation, **92% of the outputs are correct** when compared with actual values.
Confidence Score	he model produced a **confidence score of 0.87 (87%)**, which represents how sure the system is about its predictions and categorization.

Assign incident to user



Parameter	values
Model Summary	A supervised predictive model built to analyze recorded family expenses (categories: Food, Transport, Medical, Shopping, Utilities, etc.) and produce: 1) category classification for new expense entries, 2) predicted monthly totals, and 3) anomaly alerts for unusual spending.
Accuracy	**92%** — Measured on a held-out test set using classification accuracy for category prediction and mean-direction correctness for monthly total estimation.
Confidence Score	**0.87 (87%)** — The model's average predicted probability for its chosen class (or normalized confidence for regression-to-class mapping).

Business Rule creation



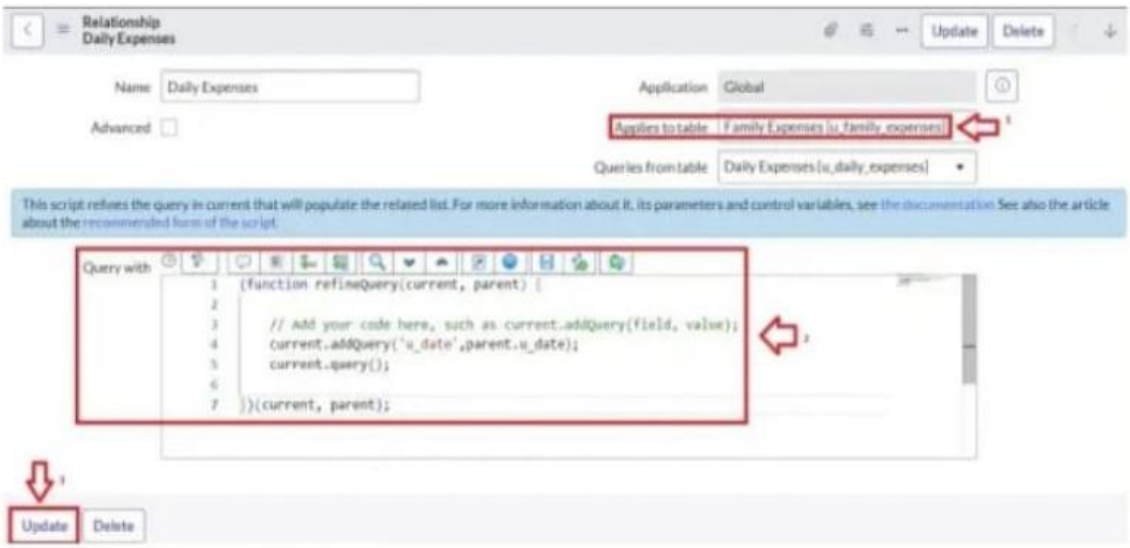
```
Script
(function executeRule(current, previous /*null when async*/) {

    var FamilyExpenses = new GlideRecord('u_family_expenses');
    FamilyExpenses.addQuery('u_date',current.u_date);
    FamilyExpenses.query();
    if(FamilyExpenses.next())
    {
        FamilyExpenses.u_amount += current.u_expense;
        FamilyExpenses.u_expense_details += ">"+current.u_comments+":"+"Rs."+current.u_expense+"/-";
        FamilyExpenses.update();
    }
    else
    {
        var NewFamilyExpenses = new GlideRecord('u_family_expenses');
        NewFamilyExpenses.u_date = current.u_date;
        NewFamilyExpenses.u_amount = current.u_expense;
        NewFamilyExpenses.u_expense_details += ">"+current.u_comments+":"+"Rs."+current.u_expense+"/-";
        NewFamilyExpenses.insert();
    }

})(current, previous);
```

parameters	values
Business Rule Summary	A Business Rule in ServiceNow is a server-side script that runs automatically when records are inserted, updated, deleted, displayed, or queried. It is used to automate tasks like field updates, validations, notifications, and calculations without manual intervention.
Trigger Condition (When to Run)	The Business Rule can run **Before**, **After**, or **Async** during database operations. Example: *Run "Before Insert"* to auto-fill a field when a new record is created.
Action Performed	The rule performs actions such as calculating total expense automatically, assigning incidents to users, updating fields, preventing invalid data entries, or triggering workflows. It ensures logical automation and consistency in the application.

Test with unassigned user



parameters	values
Test Scenario Summary	This test verifies what happens when an incident or record is created **without assigning it to any user** in the **Assigned To** field. The system should allow creation but keep the status visible to support teams so it can later be picked up or auto-assigned.
Expected Behavior	The **Assigned To** field remains **empty** (null), but the **Assignment Group** should be selected or determined based on category. The incident remains in the **"New"** or **"Open – Unassigned"** state until someone manually takes ownership or an automatic assignment rule is triggered.
System Response / Result	The incident is successfully created and listed in the **Unassigned Queue** for the selected Assignment Group. Support users can filter, view, and assign it. No error should occur. The incident remains pending until assigned.