

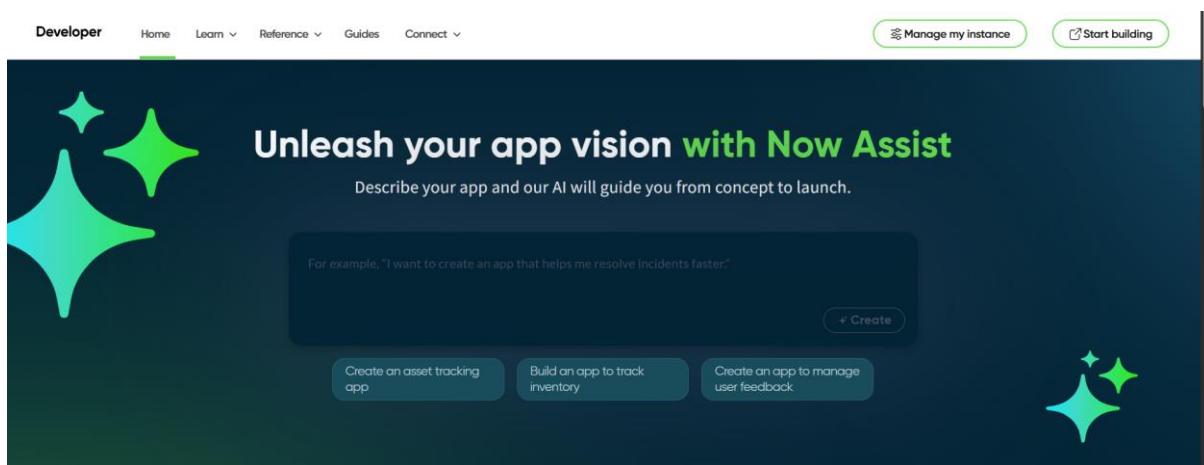
Calculating Family Expenses using Service Now

1. Setting up ServiceNow Instance

Sign up for a developer account at developer.servicenow.com.

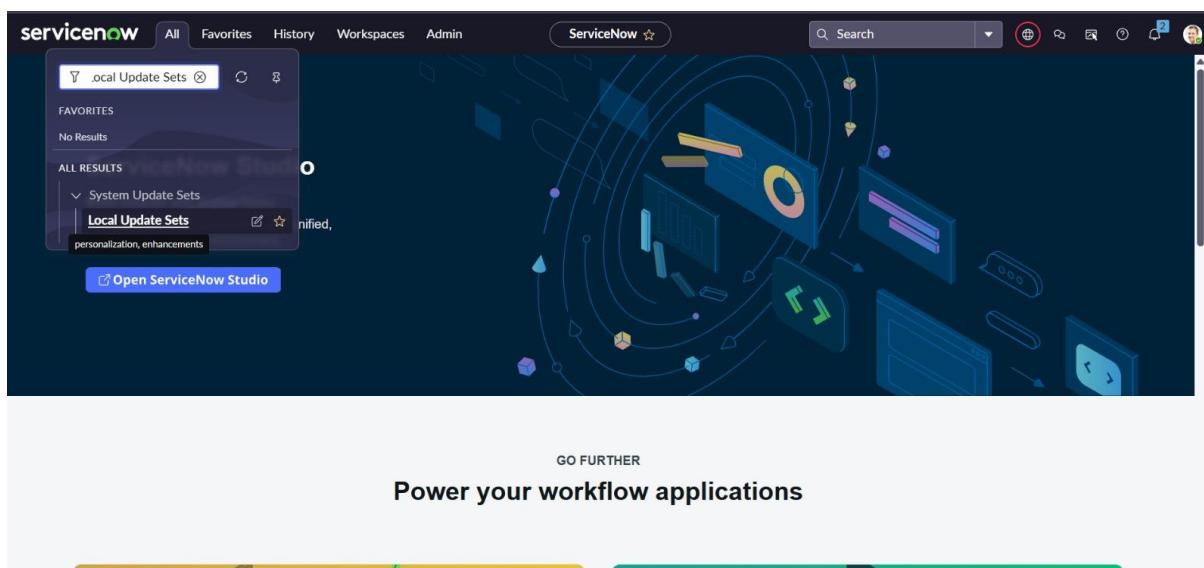
Go to Personal Developer Instance → Request Instance and fill in the required details.

Once your instance is ready, log in using the provided credentials to access ServiceNow.



2. Creation of New Update Set

In the filter navigator, search Local Update Set → click New.



Enter Name: Family Expenses, then click Submit and Make Current.

The screenshot shows the 'Update Set' interface for a record named 'Family Expenses'. The record details are as follows:

* Name	Family Expenses	Application	Global
State	In progress	Created	2025-10-29 22:10:32
Parent		Created by	admin
Release date		Merged to	
Install date			
Installed from			
Description			

At the bottom left are 'Update' and 'Delete' buttons.

3. Creation of Family Expenses Table

In the filter navigator, search Tables → click New.

Enter Label: Family Expenses, Menu Name: Family Expenditure, then right-click the header and select Save.

The screenshot shows the 'Table' creation interface for a new table named 'Family_Expenses'. The table details are as follows:

* Label	Family_Expenses	Application	Global
* Name	u_family_expenses	Remote Table	

4. Creation of Columns (Fields)

Add new rows under **Columns** with the following details:

- **Number** – String
- **Date** – Date
- **Amount** – Integer
- **Expense Details** – String (Max length: 800)

Table Family_Expenses

* Label: Family_Expenses
* Name: u_family_expenses

Application: Global

Remote Table:

Column label	Type	Reference	Max length	Default value	Display
Expense Details	String	(empty)	800		false
Updated by	String	(empty)	40		false
Sys ID	Sys ID (GUID)	(empty)	32		false
Updated	Date/Time	(empty)	40		false
Date	Date	(empty)	40		false
Amount	Integer	(empty)	40		false
Created by	String	(empty)	40		false
Number	String	(empty)	40	javascript:getNextObjNumberPadded();	false
Created	Date/Time	(empty)	40		false
Updates	Integer	(empty)	40		false

Table Columns

Column label	Type	Reference	Max length	Default value	Display
Expense Details	String	(empty)	800		false
Updated by	String	(empty)	40		false
Sys ID	Sys ID (GUID)	(empty)	32		false
Updated	Date/Time	(empty)	40		false
Date	Date	(empty)	40		false
Amount	Integer	(empty)	40		false
Created by	String	(empty)	40		false
Number	String	(empty)	40	javascript:getNextObjNumberPadded();	false
Created	Date/Time	(empty)	40		false
Updates	Integer	(empty)	40		false

Insert a new row...

Delete Update Delete All Records

Access Controls (4)

Name	Decision Type	Operation	Type	Active	Updated by	Updated
u_family_expenses	Allow If	delete	record	true	admin	2025-10-31 17:39:42
u_family_expenses	Allow If	write	record	true	admin	2025-10-31 17:39:42
u_family_expenses	Allow If	create	record	true	admin	2025-10-31 17:39:41
u_family_expenses	Allow If	read	record	true	admin	2025-10-31 17:39:42

Right-click the header and select **Save**.

4. Making Number Field an Auto-Number

Open the Number field → Advanced view.

Check Use dynamic default and select Get Next Padded Number, then click Update.

The screenshot shows the 'Dictionary Entry' screen for the 'Number' field. The top navigation bar includes 'Delete Column', 'Update', and other standard icons. The main configuration area has the following settings:

- * Table: Family_Expenses [u_family_expenses]
- * Type: String
- * Column label: Number
- * Column name: u_number
- * Max length: 40
- Application: Global
- Active: checked
- Function field: none
- Read only: unchecked
- Mandatory: unchecked
- Display: unchecked

A note below the fields states: "Alters the behavior of a field or functionality that depends on the field. [More Info](#)".

The 'Attributes' section is empty.

At the bottom, there are tabs for 'Choice List Specification', 'Calculated Value', and 'Default Value'. The 'Default Value' tab is selected, showing the configuration:

- Use dynamic default: checked
- Dynamic default value: Get Next Padded Number

Search Number Maintenance → click New.

Set Table: Family Expenses, Prefix: MFE, and click Submit.

The screenshot shows the 'Number' maintenance screen. The top navigation bar includes 'Update', 'Delete', and other standard icons. The form fields are as follows:

- * Table: Family_Expenses
- Prefix: MFE
- * Number: 1,000
- Application: Global
- Number of digits: 7

At the bottom, there are 'Update' and 'Delete' buttons, and a 'Related Links' section with a 'Show Counter' link.

5. Configure the Form

Search Family Expenses and open it.

Click New, then right-click the header → Configure → Form Design.

Arrange fields as needed.

Set Number as *Read-only* and make Date and Amount *Mandatory*.

Click Save.

The screenshot shows the 'Form Design' interface for the 'Family_Expenses [u_fam]' table. On the left, there's a sidebar with tabs for 'Fields' and 'Field Types'. Under 'Fields', there are sections for 'Filter', 'Fields' (including 'Created', 'Created by', 'Updated', 'Updated by', 'Updates'), and 'Formatters' (including 'Activities (filtered)', 'Contextual Search Results', 'Ratings'). The main area displays two sections: 'Family_Expenses [u_family_expenses]' (1 Column) containing a 'Number' field and an 'Expense Details' section, and a 'Date' field followed by an 'Amount' field in a '2 Column' section.

6. Creation of Daily Expenses Table

Search Tables → click New.

The screenshot shows the 'Table' definition interface for 'Daily_Expenses'. It includes fields for 'Label' (set to 'Daily_Expenses') and 'Name' (set to 'u_daily_expenses'). The 'Application' dropdown is set to 'Global'. A note at the top states: 'A table is a collection of records in the database. Each record corresponds to a row in a table, and each field on a record corresponds to a column on that table. Applications use tables and records to manage data and processes.' There are buttons for 'Delete', 'Update', and 'Delete All Records' at the top right.

Enter Label: Daily Expenses, add to Menu: Family Expenditure, then right-click the header and select Save.

The screenshot shows the 'Table' definition interface for 'Daily_Expenses' after saving. It includes fields for 'Label' (set to 'Daily_Expenses') and 'Name' (set to 'u_daily_expenses'). The 'Application' dropdown is set to 'Global'. A note at the top states: 'A table is a collection of records in the database. Each record corresponds to a row in a table, and each field on a record corresponds to a column on that table. Applications use tables and records to manage data and processes.' There are buttons for 'Delete', 'Update', and 'Delete All Records' at the top right. Below the table definition, a 'Columns' tab is selected, showing a list of columns with their properties:

Column label	Type	Reference	Max length	Default value	Display
Updated by	String	(empty)	40		false
Sys ID	Sys ID (GUID)	(empty)	32		false
Updates	Integer	(empty)	40		false
Expense	Integer	(empty)	40		false
Family Member Name	Reference	User	32		false
Updated	Date/Time	(empty)	40		false
comments	String	(empty)	800		false
Number	String	(empty)	40	javascript:getNextObjNumberPadded();	false
Created by	String	(empty)	40		false
Date	Date	(empty)	40		false

The screenshot shows two main sections of a software application:

- Dictionary Entries:** A table listing columns for the "Daily_Expenses" table. The columns include:

Column label	Type	Reference	Max length	Default value	Display
Updated by	String	(empty)	40		false
Sys ID	Sys ID (GUID)	(empty)	32		false
Updates	Integer	(empty)	40		false
Expense	Integer	(empty)	40		false
Family Member Name	Reference	User	32		false
Updated	Date/Time	(empty)	40		false
Comments	String	(empty)	800		false
Number	String	(empty)	40	javascript:getNextObjNumberPadded();	false
Created by	String	(empty)	40		false
Date	Date	(empty)	40		false
Created	Date/Time	(empty)	40		false

 An "Insert a new row..." button is at the bottom.
- Access Controls:** A table showing four entries for the "u_daily_expenses" table:

Name	Decision Type	Operation	Type	Active	Updated by	Updated
u_daily_expenses	Allow If	create	record	true	admin	2025-10-31 17:56:33
u_daily_expenses	Allow If	write	record	true	admin	2025-10-31 17:56:33
u_daily_expenses	Allow If	read	record	true	admin	2025-10-31 17:56:33
u_daily_expenses	Allow If	delete	record	true	admin	2025-10-31 17:56:33

7. Making Number Field an Auto-Number

Open the Number field → Advanced view.

Enable Use dynamic default and choose Get Next Padded Number, then click Update

The screenshot shows the "Advanced" view for the "Number" field in the "Daily_Expenses" table:

- Table & Column:** Set to "Daily_Expenses [u_daily_expenses]" and "Number".
- Attributes:** Application: Global, Active: checked, Function field: unchecked, Read only: checked, Mandatory: unchecked, Display: unchecked.
- Default Value:** Set to "Use dynamic default" (checked) and "Dynamic default value: Get Next Padded Number".
- Buttons:** Delete Column, Update.

Search Number Maintenance → New.

Set Table: Daily Expenses, Prefix: DFE, and click Submit.

Number DFE

* Table: Daily_Expenses

Prefix: DFE

* Number: 1,000

Application: Global

Number of digits: 7

Update Delete

Related Links
Show Counter

8. Configure the Form

Search Daily Expenses and open it.

Click New, then right-click the header → Configure → Form Design.

Arrange fields as needed.

Set Number as *Read-only* and make Date and Family Member Name *Mandatory*.

Click Save.

Default view

Form Design

Daily_Expenses [u_daily_expenses]

Fields	Field Types
Filter	
Fields	
# Created	
# Created by	
# Updated	
# Updated by	
# Updates	
Formatters	
# Activities (filtered)	
# Contextual Search Results	
# Ratings	

Number	Date
Expense	
Comments	
Member name	

9. Creating Relationship Between Tables

Search Relationships → click New.

Set Name: Daily Expenses, Applies to Table: Family Expenses, Related Table: Daily Expenses, then click Save.

Name: Daily_Expenses

Application: Global

Applies to table: Family_Expenses [u_family_expenses]

Queries from table: Daily_Expenses [u_daily_expenses]

```

Query with: Turn on ECMAScript 2021 (ES12) mode ⓘ
1 (function refineQuery(current, parent) {
2
3     // Add your code here, such as current.addQuery(field, value);
4     current.addQuery('u_date',parent.u_date);
5     current.query();
6
7 })(current, parent);

```

Run Query Diagnostics | Update | Delete

10. Configuring Related List

Open Family Expenses → New → Configure > Related Lists → add Daily Expenses → Save.

Available: Attachments

Selected: Daily_Expenses

View name: Default view

Related Links:

- Show versions
- Related list performance diagnostics

Cancel | Save

11. Business Rule Creation

Navigate to All → Business Rules → New.

Name: Family Expenses BR

Table: Daily Expenses

Add query: required

Business Rule
Family Expenses BR

A business rule is a server-side script that runs when a record is displayed, inserted, deleted, or when a table is queried. Use business rules to automatically change values in form fields when the specified conditions are met. [More Info](#)

Name: Family Expenses BR Application: Global Active: Advanced:

Table: Daily_Expenses [u_daily_expenses] When: before Order: 100

When to run Actions Advanced

Specify whether the business rule should run on Insert or Update. Use Filter Conditions to specify under which conditions the business rule should run.

When: before Order: 100

Insert: Update: Delete: Query:

Filter Conditions: Add Filter Condition, Add OR Clause

-- choose field -- -- oper -- -- value --

```

Condition: Turn on ECMAScript 2021 (ES12) mode

Script:
 Turn on ECMAScript 2021 (ES12) mode
 Turn off ECMAScript 2021 (ES12) mode

1 (function executeRule(current, previous /*null when async*/) {
2
3
4 var FamilyExpenses = new GlideRecord('u_family_expenses');
5
6 FamilyExpenses.addQuery('u_date',current.u_date);
7
8 FamilyExpenses.query();
9
10 if(FamilyExpenses.next())
11 {
12
13 FamilyExpenses.u_amount += current.u_expense;
14 FamilyExpenses.u_expense_details += ">" + current.u_comments + ":" + "Rs." + current.u_expense + "-";
15 FamilyExpenses.update();
16
17 }
18 else
19
20
21
22
23

```

12. Configure Relationship

Go to All → Relationships and open Daily Expenses Relationship.
Set Applies to table: Family Expenses.

Add Query:

```
(function refine Query (current, parent) {

    current.add Query('update', parent.u_date);

    current.query ();

}) (current, parent);
```

Query with Turn on ECMAScript 2021 (ES12) mode 

```
(function refineQuery(current, parent) {  
  // Add your code here, such as current.addQuery(field, value);  
  current.addQuery('u.date',parent.u_date);  
  current.query();  
})(current, parent);
```

[Run Query Diagnostics](#) [Update](#) [Delete](#)

Update the Relationships by clicking on update.

- Built in **ServiceNow** to manage family spending.
 - **Daily Expenses** table records:
 - Date, Amount, Comments, Family Member
 - **Family Expenses** table shows:
 - Total spent per date, with summary
 - Tables are **linked**:
 - Daily entries update family totals automatically
 - Helps track spending and keep financial records organized.

Daily_Expenses					Number	Search	Actions on selected rows...	New
All								
	Number	comments	Date	Expense	Family Member Name			
	DFE0001003	money	2025-10-31	500	Abraham Lincoln			
	DFE0001002	Mobile	2025-11-01	500	Abel Tuter			

	Family_Expenses	Number	Search	Actions on selected rows...	New
All					
	Number	Amount	Date	Expense Details	
	MFE0001011	1,000	2025-10-31	new	
	MFE0001009	500	2025-11-01	Mobile	

The screenshot shows two main parts of a ServiceNow application interface.

Record Creation Page:

- Header:** Family_Expenses, MFE0001009
- Fields:**
 - Number: MFE0001009
 - Expense Details: Mobile
 - Date: 2025-11-01
 - Amount: 500
- Buttons:** Update, Delete

List View:

- Header:** Daily_Expenses, Number
- Search Bar:** Search
- Actions:** Actions on selected rows..., New
- Table:**| | Number | comments | Date | Expense | Family Member Name |
| --- | --- | --- | --- | --- | --- |
| | DFE0001002 | Mobile | 2025-11-01 | 500 | Abel Tuter |
- Pagination:** 1 to 1 of 1

Conclusion:

In conclusion, the *Family Expenses Calculation System* built on ServiceNow provides an efficient and organized way to manage household finances. By leveraging ServiceNow's automation and data management capabilities, the system simplifies expense tracking, ensures accuracy, and offers real-time insights into family spending patterns. This project not only enhances financial transparency but also promotes better budgeting and informed decision-making—ultimately contributing to improved financial stability and well-being for families.

Done By,

Team ID : NM2025TMID06562

Team Size : 4

Team Leader : Rosa Mystica M

Team member : Sahaya Shiny Vaz A

Team member : Santhiya M

Team member : Snowfa P Rayer J

Thank You!