

CALCULATING FAMILY EXPENSES USING SERVICE NOW

NAAN MUDALVAN PROJECT
TEAM ID:

Team Leader : SAJIN S

Team member : SELVA HARINIS

Team member : SANTHIYA D

Team member : SANTHOSH KUMAR M

ST.JOSEPH'S COLLEGE (ARTS & SCIENCE)

INDEX

1. Introduction 2. Abstract 3. Problem statement 4. Solution 5. Practical use 6. Knowledge gained 7. Milestone 1: Setting up service now instance 8. Milestone 2: Creation of new update set 9. Milestone 3: Creation of table family expenses
10. Milestone 4: Creation of table daily expenses
11. Milestone 5: Creation of relationship
12. Milestone 6: Configuring Related list on family expenses
13. Milestone 7: Creation of Business rules
14. Milestone 8: Configure the relationship
15. Conclusion

INTRODUCTION

Managing family expenses is one of the most important aspects of financial planning, yet it is often overlooked or handled in unstructured ways like handwritten notes or spreadsheets. Families frequently face challenges such as overspending, lack of budget control, and difficulty in analyzing where their money goes. To address these issues, technology can play a vital role by providing structured, automated, and user-friendly solutions.

This project, "Calculating Family Expenses Using ServiceNow," aims to create a digital system that helps track, categorize, and manage family expenses efficiently. ServiceNow, known for its robust workflow automation and application development capabilities, provides an excellent platform for building such a system. By leveraging ServiceNow, this project transforms expense management into a more streamlined process with features like expense categorization, daily tracking, budget limits, reporting, and automated business rules.

The system will not only simplify financial management for families but also provide real-time insights into spending patterns, allowing better decision-making. With its scalability and flexibility, the solution can be expanded to suit different family structures and even adapted for small business expense management. Ultimately, the project showcases how a powerful enterprise platform like ServiceNow can be applied beyond IT workflows to solve everyday problems in an innovative and practical way.

ABSTRACT

Expense management plays a crucial role in maintaining financial stability within families, yet traditional methods such as manual tracking or spreadsheets often lead to errors, lack of visibility, and poor decision-making. To address this challenge, this project focuses on designing and implementing a Family Expense Management System using ServiceNow.

The application leverages ServiceNow's powerful low-code development environment to create structured tables, relationships, and automated workflows for recording and analyzing expenses. Key features include categorizing expenses (such as food, utilities, and transport),

maintaining family member details, linking daily transactions, and applying business rules to validate data and automate calculations. By configuring related lists and creating meaningful reports, the system provides users with a clear overview of their financial habits and budget limits.

The proposed solution not only simplifies day-to-day

expense

tracking but also provides real-time insights into spending patterns, enabling families to make smarter financial decisions. Furthermore, the project demonstrates the versatility of ServiceNow beyond traditional IT service management, showcasing its potential in solving practical, real-world problems.

PROBLEM STATEMENT

Managing household expenses is often a difficult and time-consuming task for families. Most families rely on manual methods such as notebooks, receipts, or spreadsheets to track their daily spending. These methods come with several challenges:

- Lack of real-time tracking of expenses.
- Difficulty in categorizing and consolidating expenses like food, utilities, rent, and transportation.
- Limited ability to analyze spending patterns or generate reports.
- High chances of errors due to manual data entry.
- No automation to alert families about overspending or exceeding budgets.

As a result, families often lose visibility into their financial flow, making it harder to control budgets or make informed

financial decisions. This creates the need for a systematic, automated, and user-friendly solution to manage and calculate

family expenses efficiently.

SOLUTION

The proposed solution is to build a Family Expense Management System on the ServiceNow platform. ServiceNow, being a robust low-code/no-code platform, provides all the tools required to create structured applications without needing extensive programming knowledge.

The solution involves:

- Creating custom tables to store family member details and daily expenses.
- Defining relationships between family members and their respective expenses for easy tracking.
- Configuring related lists so that expenses linked to each family member can be viewed in one place.
- Implementing business rules to automate calculations, validate entries, and ensure data accuracy.
- Generating reports and dashboards to visualize monthly/annual spending and highlight budget deviations.
- Using update sets to track and migrate customizations, ensuring proper version control.

PRACTICAL USE

The Family Expense Management System built on ServiceNow helps families track, categorize, and analyze their expenses in a structured way. It simplifies budgeting, reduces errors from manual tracking, and provides real-time insights into spending patterns. Families can use it to set limits, monitor monthly expenses, and generate reports for better financial decisions. Beyond households, the same system can be adapted for small businesses to manage cash flow and daily transactions effectively, proving the versatility of ServiceNow in solving practical, non-IT problems.

KNOWLEDGE GAINED

- Learned how to set up and configure a ServiceNow developer instance for building applications. Understood the importance of update sets for tracking and migrating customizations.
- Gained practical skills in creating custom tables and defining fields to store structured data.
- Learned how to establish relationships between tables for linked data management.
- Practiced configuring related lists for easier navigation and record visibility.
- Understood how to create and apply business rules for automation and validations.
- Gained insights into data modeling and database concepts within ServiceNow.
- Learned how to generate reports and dashboards for real-time analysis.
- Understood how ServiceNow can be applied to non-IT use cases like family expense tracking.
- Improved overall knowledge of workflow automation and low-code development.

MILESTONE 1: SETTING UP THE SERVICE NOW INSTANCE

- Go to the official ServiceNow Developer portal: <https://developer.servicenow.com> and create a developer account.
- After signing in, open the Personal Developer Instance section from the dashboard.
- Select Request Instance to generate a fresh ServiceNow environment for development.
- Provide the necessary details (like version selection) and confirm your request.
- Wait for the confirmation email containing your instance URL and login credentials.
- Use the credentials to log in to your newly created ServiceNow instance.
- Once inside, explore the interface and begin working on the platform.

The screenshot shows the ServiceNow developer portal interface. At the top, there's a navigation bar with links for MyNow, Products, Industries, Learning, Support, Partners, and Company. Below that is a sub-navigation bar for the 'Developer' section, with links for Home, Learn, Reference, Guides, and Connect. On the right side of the header, there are buttons for 'Manage my instance' and 'Start building'. The main content area is titled 'Manage my instance (dev221986)'. It features three main cards: 1) A status card showing 'Status Online' with three studios listed: App engine studio (Installed), Creator studio (Installed), and ServiceNow studio (Installed). It also shows the current version as Xanadu and a 'Upgrade release' button. 2) An instance configuration card with fields for 'Instance URL' (https://dev221986.service-now.com), 'User name' (admin), 'Current password' (redacted), and 'User role' (Admin). 3) A 'Useful links' card with links to the PDI Guide, FAQs, Managing your PDI, and the Developer advocate blog. At the bottom, there's a section for 'Plugins for your instance (53)' with buttons for 'All activation statuses' and 'All demo data statuses', and a search bar.

MILSTONE 1: CREATION OF NEW UPDATE SET

- Log in to your ServiceNow instance and go to the Application Navigator.
- Search for Update Sets and open Local Update Sets under System Update Sets.
- Click on New to create a fresh update set.
- Enter the following details:
 - Name: Family Expenses
 - Description: Update set to capture all configurations related to the Family Expense Management project.
- Save the record and mark it as the Current Update Set, so every change you make is tracked under this set.
- Verify that the update set is active by checking the header at the top of the screen.
- From this point forward, all customizations (tables, relationships, and business rules) will be recorded inside the Family Expenses update set.

The screenshot shows the 'Update Set - Create New Update Set' page in ServiceNow. The 'Name' field is populated with 'Family Expenses'. The 'State' dropdown is set to 'In progress'. The 'Parent' field has a dropdown arrow. The 'Release date' field contains a calendar icon. The 'Description' field is empty. At the bottom, there are two buttons: 'Submit' and 'Submit and Make Current'.

MILSTONE 1: CREATION OF TABLE FAMILY EXPENSES

Activity 1 - Creating the Family Expenses Table

- . In your ServiceNow instance, navigate to All < Tables using the filter navigator.
- . Click on New to create a new table.
- . Fill in the required details:
 - o Label: Family Expenses
 - o Name: (This will be auto-generated based on the label)
 - o New Menu Name: Family Expenditure
- . Save the record to create the new table.

The screenshot shows the ServiceNow 'Table - New Record' page. At the top, there are tabs for 'All', 'Favorites', 'History', 'Workspaces', and 'Admin'. The title bar says 'Table - New Record'. Below the title bar, there's a search bar and a 'Submit' button. A message box says 'ServiceNow recommends creating custom tables in scoped applications. To learn more about creating scoped applications, click [here](#)'. A help box explains what a table is. The main form has fields for 'Label' (Family Expenses), 'Name' (u_family_expenses), 'Extends table' (dropdown menu), 'Application' (Global), 'Create module' (checkbox checked), 'Create mobile module' (checkbox checked), 'Add module to menu' (dropdown menu), and 'New menu name' (Family Expenses). At the bottom, there are tabs for 'Columns', 'Controls', and 'Application Access', and a table titled 'Dictionary Entries' with columns for 'Column label', 'Type', 'Reference', 'Maxlength', 'Default value', and 'Display'. A note at the bottom says 'Insert a new row..'

MILSTONE ✓: CREATION OF TABLE FAMILY EXPENSES

Activity ✓ - Adding Columns to the Family Expenses Table

Duration: 1 Hour

Skill Tags: Table Configuration, Data Modeling, ServiceNow Basics

- Open the newly created Family Expenses table.
- To add columns, double-click near the existing columns to insert a new row.
- Enter the following details one by one:
 - ↳ Column Label: Number
 - ↳ Type: String
 - ↳ Column Label: Date
 - ↳ Type: Date
 - ↳ Column Label: Amount
 - ↳ Type: Integer
 - ↳ Column Label: Expense Details
 - ↳ Type: String
 - ↳ Max Length: 100

The screenshot shows the ServiceNow interface for managing the 'Family Expenses' table. At the top, there's a navigation bar with links for All, Favorites, History, Workspaces, and Admin. The title 'Table - Family Expenses' is displayed. Below the title, there are buttons for Search, Delete, Update, and Delete All Records. The main area is a grid showing table columns. A search bar at the top of the grid allows filtering by 'Table Columns' or 'for text'. The grid has columns for 'Dictionary Entries', 'Column label', 'Type', 'Reference', 'Max length', 'Default value', and 'Display'. There are seven rows in the grid, each with a delete icon. The last row, 'Expense', has its 'Type' field set to 'String' and is highlighted with a blue selection bar. A note at the bottom of the grid says 'Insert a new row...'. The status bar at the bottom indicates '1 to 6 of 6'.

Dictionary Entries	Column label	Type	Reference	Max length	Default value	Display
	Updates	Integer	{empty}	40		false
	Updated	Date/Time	{empty}	40		false
	Created by	String	{empty}	40		false
	Sys ID	Sys ID (GUID)	{empty}	32		false
	Updated by	String	{empty}	40		false
	Created	Date/Time	{empty}	40		false
X	Number	String				false
X	Date	Date				false
X	Amount	Integer				false
X	Expense	String				false

MILSTONE ↴: CREATION OF TABLE FAMILY EXPENSES

Activity ↴ - Making the Number Field an Auto-Number
Open the Family Expenses table.

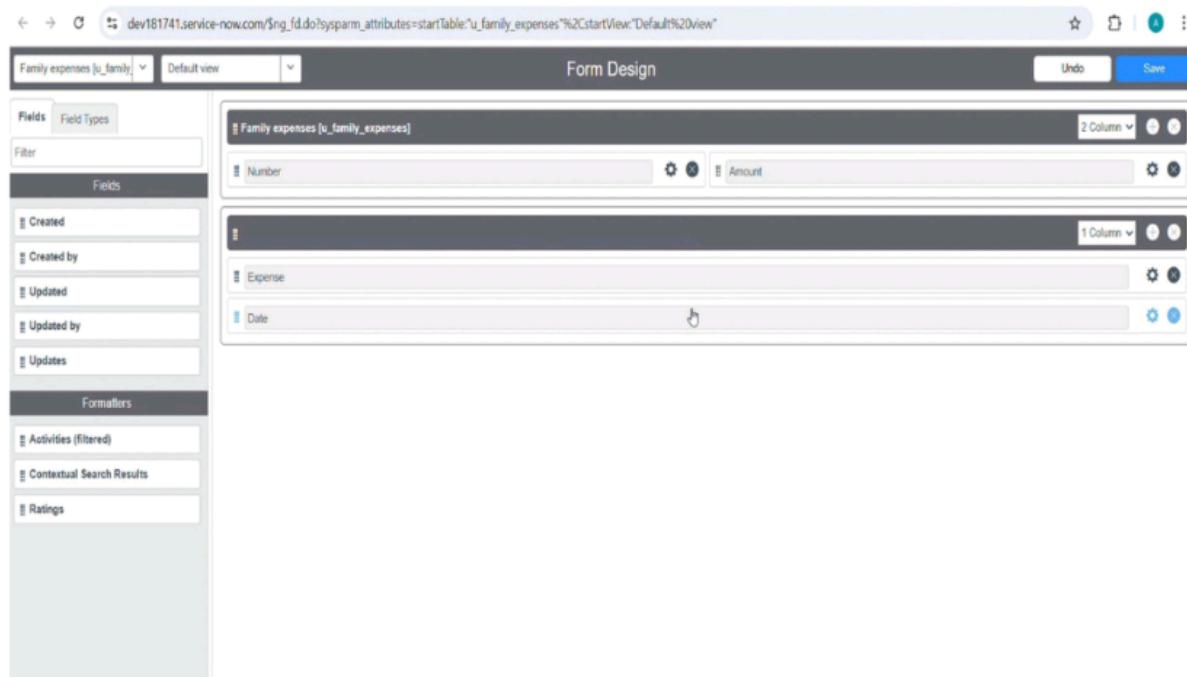
- Locate the Number field /column and double-click to open its properties.
- Scroll down and switch to the Advanced view.
- In the Default Value section:
 - Check the box for Use Dynamic Default.
 - Set the Dynamic Default Value to Get Next Padded Number.
- Click Update to save the changes.

The screenshot shows the ServiceNow dictionary entry for a 'Number' field. The URL in the browser is `dev221906.service-now.com/nav/uiclassic/params/target/sys_dictionary.do?sys_id=30e8340a2583732210639fc39feaad3ba%26sysparm_view=Advanced`. The page title is 'Dictionary Entry - Number'. The top navigation bar includes links for All, Favorites, History, Workspaces, and a search bar. Below the navigation is a toolbar with icons for Delete Column and Update. The main content area shows the 'Number' field's properties. The 'Max length' is set to 40. The 'Attributes' section is empty. The 'Default Value' tab is selected, showing the configuration for dynamic default values. The 'Use dynamic default' checkbox is checked, and the 'Dynamic default value' dropdown is set to 'Get Next Padded Number'. Buttons for Delete Column and Update are at the bottom of this section. A 'Related Links' section is visible at the very bottom.

MILSTONE 4: CREATION OF TABLE FAMILY EXPENSES

Activity 4 – Configuring the Form

- Navigate to All < in the filter, search for Family Expenses.
- Open the Family Expenses table.
- Click on New to create a new form entry.
- On the form header, right-click and select:
 - Configure < Form Design.
- In the Form Designer, use drag-and-drop to:
 - Rearrange fields.
 - Group related fields together.
 - Add sections if required for better clarity.
- Save the customized form layout.



MILSTONE 1 : CREATION OF TABLE DAILY EXPENSES

Activity 1 - Creating The Daily Expenses Table

- Navigate to All < Tables using the filter navigator .
- Click on New to create a new table .
- Fill in the required details :
 - Label : Daily Expenses
 - Name : (Auto-populated by the system)
 - Add Module to Menu : Family Expenditure
 - Go to the form header , right-click , and select Save .

The screenshot shows the ServiceNow 'Table - New Record' page. At the top, there are tabs for 'All', 'Favorites', 'History', 'Workspaces', and 'Admin'. The main title is 'Table - New Record'. Below the title, there is a message from ServiceNow recommending creating custom tables in scoped applications. A blue banner at the bottom provides information about tables and records.

Table Configuration Fields:

- * Label: Daily Expenses
- * Name: u.daily_expenses
- Extends table: (empty)
- Application: Global
- Create module:
- Create mobile module:
- Add module to menu: --Create new--
- New menu name: Daily Expenses
- Remote Table:

Dictionary Entries:

Column label	Type	Reference	Max length	Default value	Display
Table Columns	for text				

MILSTONE 1: CREATION OF TABLE DAILY EXPENSES

Activity 1 – Creating Columns (Fields)

- Open the Daily Expenses table.
- Near Columns, double-click to insert a new row and add the following fields:
 - ↳ 1. Column Label: Number
 - Type: String
 - ↳ 2. Column Label: Date
 - Type: Date
 - ↳ 3. Column Label: Expense
 - Type: Integer
 - ↳ 4. Column Label: Family Member Name
 - Type: Reference
 - Max Length: 100
 - ↳ 5. Column Label: Comments
 - Type: String
 - Max Length: 1000

The screenshot shows the ServiceNow Table - Daily Expenses page. The table has the following columns and data:

Column Label	Type	Default Value	Length	Required
Updated by	String	(empty)	40	false
Updates	Integer	(empty)	40	false
Updated	Date/Time	(empty)	40	false
Created by	String	(empty)	40	false
Created	Date/Time	(empty)	40	false
Sys ID	Sys ID (GUID)	(empty)	32	false
Number	String			false
Date	Date			false
Expense	Integer			false
Family member	Reference			false

At the bottom of the table, there is a link "Insert a new row..." and buttons for Delete, Update, and Delete All Records. Below the table, there is a "Related Links" section with links to Form Builder, Design Form, Layout Form, Layout List, Show Form, and a copyright notice for © 2014 ServiceNow Inc.

MILSTONE 1 : CREATION OF TABLE DAILY EXPENSES

Activity 1 – Making Number Field an Auto-Number

- Open the Daily Expenses table.
- Locate the Number field /column and double-click to open its properties.
- Scroll down and switch to the Advanced View.
- In the Default Value section:
 - Check the box for Use Dynamic Default.
 - Set the Dynamic Default Value to Get Next Padded Number.
- Click Update to save changes.

Configuring Number Maintenance:

- Navigate to All < Number Maintenance.
- Click on New.
- Enter the details as follows:
 - Table: Family Expenses
 - Prefix: MFE
- Click on Submit.

The screenshot shows the ServiceNow interface for 'Dictionary Entry - Number'. A new entry is being created under 'Number View: Advanced'. The 'Default Value' tab is selected, showing the configuration for a dynamic default value. The 'Use dynamic default' checkbox is checked, and the 'Dynamic default value' field contains 'Get Next Padded Number'. Other tabs like 'Choice List Specification' and 'Calculated Value' are also visible. Below the tabs, there's a note about the default value specification. At the bottom, there are buttons for 'Delete Column' and 'Update', along with a 'Related Links' section containing 'Show Table', 'Run Point Scan', and 'Default view'. A search bar and a table for 'Access Controls' are also present at the bottom.

Number - Create DFE

Number

Invalid insert

* Table Daily Expenses

Prefix DFE

* Number 1,000

Application Global

Number of digits 7

Submit

Related Links

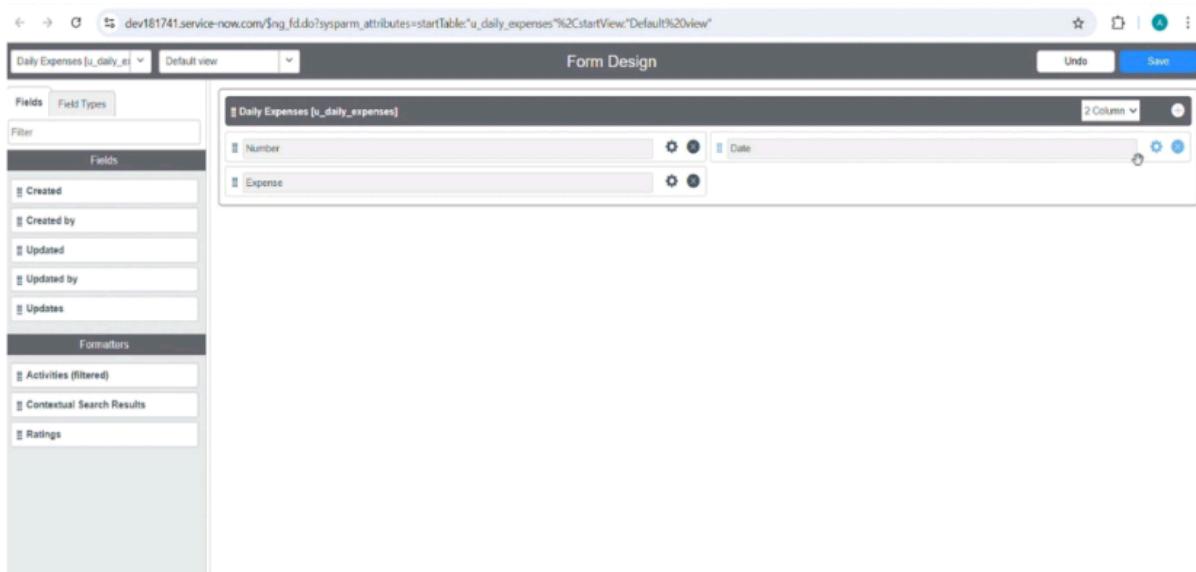
Show Counter

This screenshot shows a ServiceNow interface for creating a new number record. The title bar indicates the page is 'Number - Create DFE'. The main form has several fields: 'Table' set to 'Daily Expenses', 'Prefix' set to 'DFE', 'Number' set to '1,000', 'Application' set to 'Global', and 'Number of digits' set to '7'. A red error message 'Invalid insert' is displayed above the form. Below the form, there is a 'Submit' button and a 'Related Links' section containing a single link 'Show Counter'.

MILSTONE 1 : CREATION OF TABLE DAILY EXPENSES

Activity 1 – Configure The Form

- Navigate to All < Daily Expenses using the filter.
- Open the Daily Expenses table.
- Click on New to create a new form entry.
- On the form header, right-click, then select:
- Configure < Form Design.
- In the Form Designer, drag and drop fields to customize the form layout as per requirement.
- Apply the following configurations:
 - Number Field → Set as Read-Only by clicking the gear icon and checking Read-Only.
 - Date Field → Set as Mandatory by clicking the gear icon and checking Mandatory.
 - Family Member Name Field → Set as Mandatory using the same method.
- Click Save to apply the changes.



MILSTONE 5: CREATION OF RELATIONSHIP BETWEEN FAMILY EXPENSES AND DAILY EXPENSES TABLES

- Navigate to All < Relationships using the filter navigator.
- Click on New to create a new relationship.
- Fill in the details as follows:
 - Name: Daily Expenses
 - Applies to Table: Family Expenses
 - Related List Table: Daily Expenses
- Click Save.

The screenshot shows the ServiceNow interface for creating a new relationship. The URL in the address bar is `dev181741.service-now.com/nav/uiclassic/params/target/sys_relationship.do?sys_id=30a834e116c3fb6210555b3942b4013151%26sysparm_view=3D%26sysparm_domain=%3Dnull%2...`. The top navigation bar includes links for All, Favorites, History, Workspaces, and a search bar. The main title is "Relationship - Daily Expenses". The form fields are as follows:

Name	Daily Expenses	Application	Global
Advanced	<input type="checkbox"/>	Applies to table	Family expenses [u.family_expenses]
		Queries from table	Daily Expenses [u.daily_expenses]

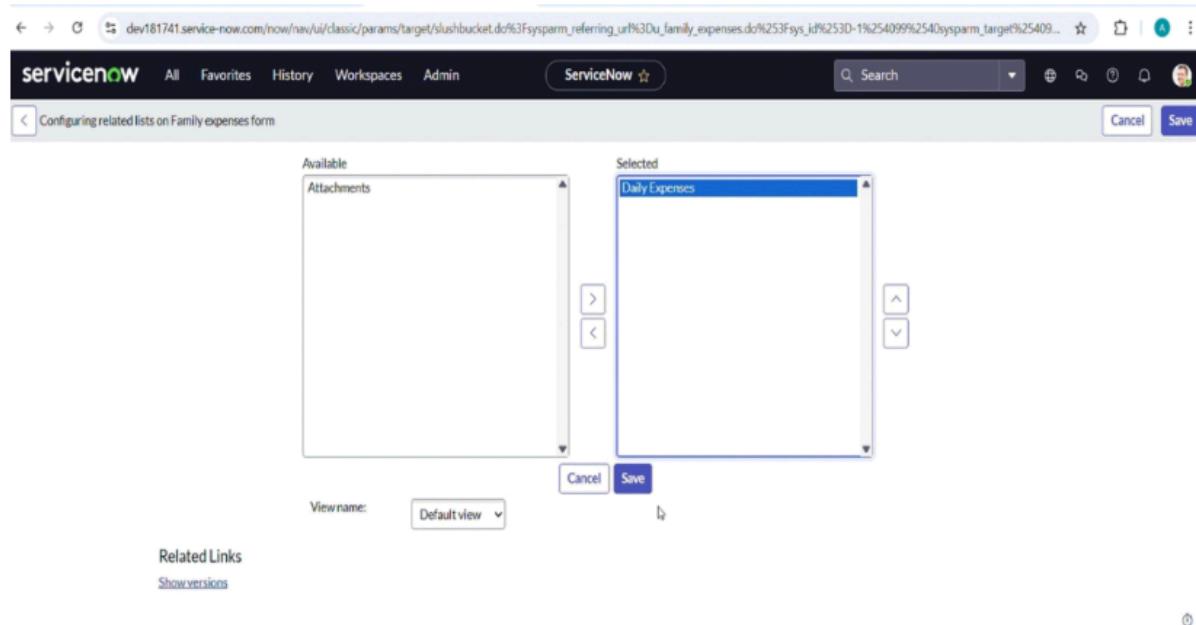
A note below the form states: "This script refines the query in current that will populate the related list. For more information about it, its parameters and control variables, see the documentation See also the article about the recommended form of the script." A code editor window shows a snippet of ECMAScript 2021 (ES12) mode code:

```
1 (function refineQuery(current, parent) {  
2     // Add your code here, such as current.addQuery(field, value);  
3 })(current, parent);
```

At the bottom of the page are "Update" and "Delete" buttons, along with "Related Links" and "Run Point Scan" links.

MILSTONE 1: CONFIGURING RELATED LIST ON FAMILY EXPENSES

- Navigate to All < Family Expenses using the filter.
- Open the Family Expenses table.
- Click on New to open the form view.
- On the form header, right-click, then select:
- Configure < Related Lists.
- From the available options, add Daily Expenses to the Selected Area.
- Click Save to apply the changes.



MILSTONE v: CREATION OF BUSINESS RULES

- Navigate to All Business Rules using the filter.
- Under System Definition, select Business Rules and click New.
- Enter the following details:
 - Name: Family Expenses BR
 - Table: Daily Expenses
 - Check Advanced.
- In the When to run section, check:
 - Insert
 - Update
- In the Advanced tab, add the following 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):

Go to the form header, right-click, then select Save.

The screenshot shows the ServiceNow interface for creating a new business rule. The top navigation bar includes links for All, Favorites, History, Workspaces, and a search bar. The main title is "Business Rule - New Record". Below the title, there are fields for "Name" (Family expenses BR), "Table" (Daily Expenses [u_daily_expenses]), "Application" (Global), and "Active" (checked). The "Advanced" tab is selected. In the "Script" section, the "Turn on ECMAScript 2021 (ES12) mode" option is checked. The script editor contains the following code:

```
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.hasNext()) {  
11 FamilyExpenses.next();  
12 FamilyExpenses.setInsert(true);  
13 FamilyExpenses.insert();  
14 }  
15 } );
```

MILSTONE 1: CONFIGURE THE RELATIONSHIP

- Navigate to All < Relationships using the filter navigator.
- Open the existing Daily Expenses Relationship.
- Update the details as follows:
 - Applies to Table: Family Expenses
- In the Query with section, enter the following script:

```
(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);
```

Click Update to save the configuration

The screenshot shows the ServiceNow interface for configuring a relationship. The top navigation bar includes links for All, Favorites, History, Workspaces, and a search bar. The main title is "Relationship - Daily Expenses". Below the title, there are fields for "Name" (set to "Daily Expenses"), "Advanced" (unchecked), "Application" (set to "Global"), "Applies to table" (set to "Family expenses [u_family_expenses]"), and "Queries from table" (set to "Daily Expenses [u_daily_expenses]"). A note at the bottom states: "This script refines the query in current that will populate the related list. For more information about it, its parameters and control variables, see [the documentation](#). See also the article about the recommended form of the script." Below this note is a code editor window titled "Query with" containing the following script:

```
4 // Add your code here, such as current.addQuery(field, value);  
5  
6 current.addQuery('u_date', parent.u_date);  
7  
8 current.query();  
9  
10 })(current, parent);
```

At the bottom of the page are "Update" and "Delete" buttons, and a "Related Links" section.

CONCLUSION

The Family Expenses Management System built on ServiceNow demonstrates how the platform can be leveraged beyond IT workflows to solve real-world problems. By systematically creating tables, relationships, forms, and business rules, the project enables seamless tracking of both daily expenses and family-level expenses in an automated manner.

The use of auto-numbering, mandatory fields, related lists, and business rules ensures data integrity, consistency, and accuracy. The relationship configuration further enhances visibility by linking daily records to family-level summaries, providing a clear financial overview.

Through this project, we learned how to apply ServiceNow features such as table creation, form design, field configuration, scripting, and automation to build a complete application. More importantly, it highlights how low-code/no-code platforms like ServiceNow can be extended into personal finance, household management, and non-IT use cases.

Overall, the project provides a practical, scalable, and user-friendly solution for managing family expenses efficiently while also strengthening skills in ServiceNow application development.