

Calculating Family Expenses Using ServiceNow...

Category: ServiceNow

Team Members: Hemanth

: Ravi Shankar Jakki

:Pavan

:Anusha

Institution: Aditya College Of Engineering & Technology

Mentor: Anji babu sir

Project Overview

The goal of this guided project was to develop a simple application in ServiceNow to help track and manage daily family expenses. It includes custom tables, form configurations, and a relationship between family members and their associated expenses. This project was done as part of the SmartInternz platform in collaboration with ServiceNow.

Tools & Platform Used

- ServiceNow Personal Developer Instance (PDI)
- Update Sets for versioning
- Tables, Forms, and Relationships
- Business Rules (optional)
- SmartInternz Guided Dashboard

Features Implemented

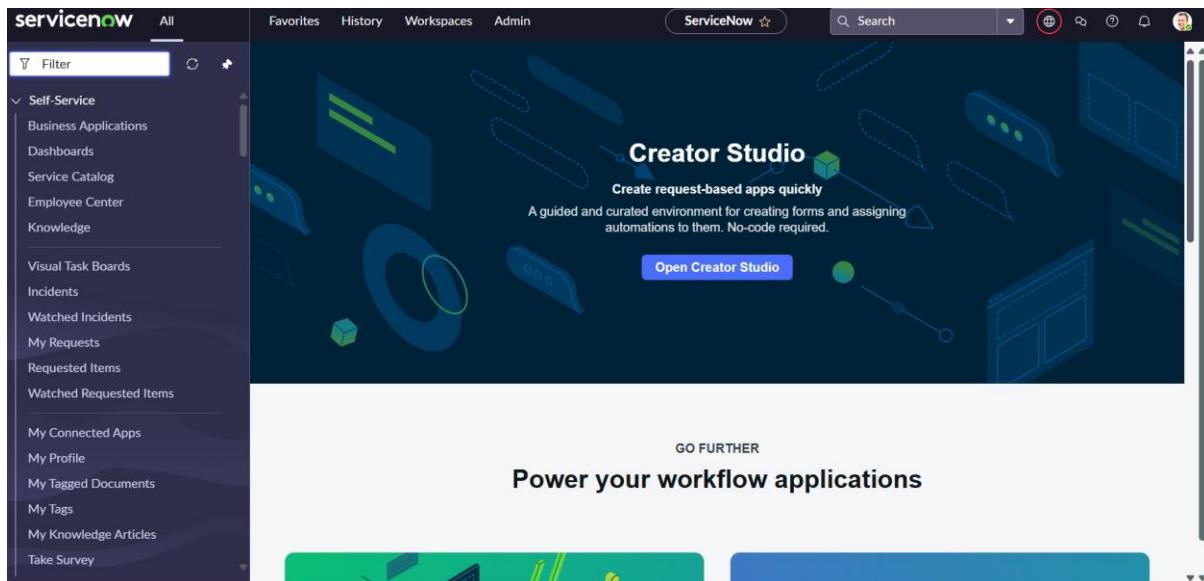
- Custom application: *Family Expenses*
- Created a custom table for *Family Expenses*
- Created a custom table for *Daily Expenses*

Calculating Family Expenses Using ServiceNow...

- Configured a one-to-many relationship between Family Members and Expenses
- Captured all changes in an Update Set
- Exported .xml file for submission

Design & Implementation:

1. Setting up service now instance:



2. Creation of New Update Set

Calculating Family Expenses Using ServiceNow...

The screenshot shows the ServiceNow interface for managing update sets. The top navigation bar includes 'Favorites', 'History', 'Workspaces', and 'Admin'. The title bar says 'Update Set - Family Expenses'. The main form contains fields for 'Name' (Family Expenses), 'State' (In progress), 'Parent' (empty), 'Release date' (empty), 'Install date' (empty), 'Installed from' (empty), and 'Description' (empty). To the right, details like 'Application' (Global), 'Created' (2025-06-22 10:51:54), 'Created by' (admin), and 'Merged to' (empty) are listed. Below the form is an 'Update' button. A 'Related Links' section includes 'Merge With Another Update Set' and 'Scan Update Set'. At the bottom, tabs for 'Customer Updates (140)', 'Update Set Logs', 'Child Update Sets', and 'Install History' are visible, along with a search bar and an 'Actions on selected rows...' dropdown.

Tables Creation:

Table 1: Family Expenses Table

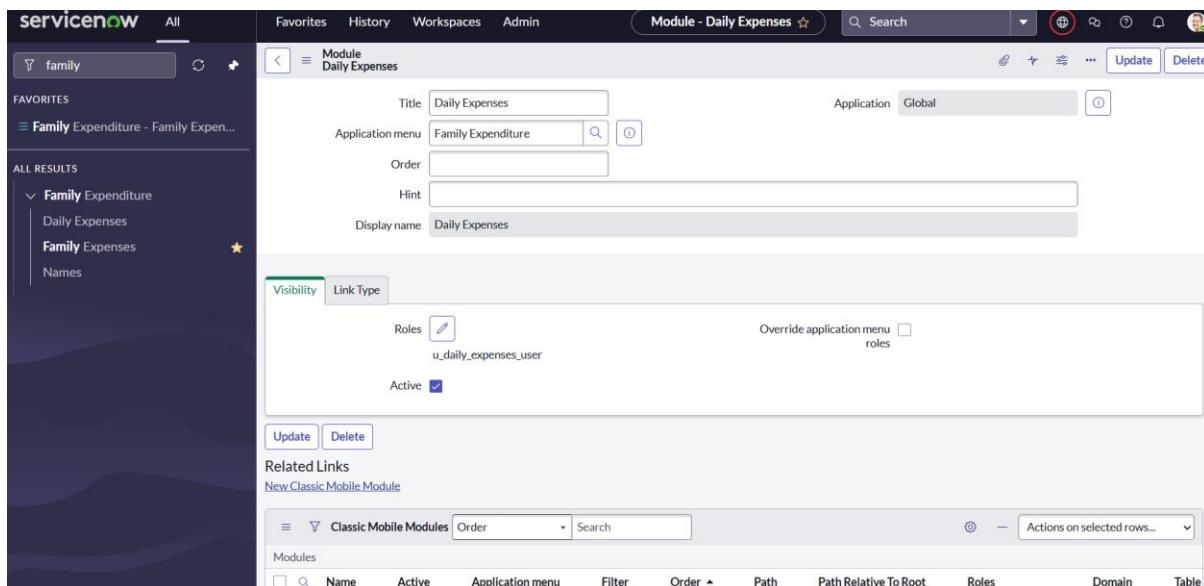
- o Fields:
 - Number
 - Date
 - Amount
 - Expenses Details

The screenshot shows the ServiceNow interface for creating a new module. The top navigation bar includes 'Favorites', 'History', 'Workspaces', and 'Admin'. The title bar says 'Module - Family Expenses'. The main form contains fields for 'Title' (Family Expenses), 'Application menu' (Family Expenditure), 'Order' (empty), 'Hint' (empty), and 'Display name' (Family Expenses). Below the form is a 'Visibility' tab, which includes a 'Link Type' section with 'Roles' (u_family_expenses_user) and an 'Override application menu roles' checkbox. An 'Active' checkbox is checked. At the bottom, there are 'Update' and 'Delete' buttons. A 'Related Links' section includes 'New Classic Mobile Module'. At the very bottom, tabs for 'Classic Mobile Modules', 'Order', 'Search', and an 'Actions on selected rows...' dropdown are visible.

Calculating Family Expenses Using ServiceNow...

Table 2: Daily Expenses

- Date
- Number
- Expenses Type
- Comments
- Family Member Name



3. Relationship:

- One-to-Many: One family member can have multiple expenses
- Implemented using Reference fields and Related Lists

4. Creation Of Business Rules:

Calculating Family Expenses Using ServiceNow...

The screenshot shows the ServiceNow interface for configuring a business rule. The left sidebar navigation includes 'business rules' under 'FAVORITES'. The main content area is titled 'Business Rule - Family Expenses BR'. It displays the rule's name ('Family Expenses BR'), application ('Global'), and status ('Active'). The 'When to run' tab is selected, showing options for 'before' (Insert checked, Update checked, Delete unchecked, Query unchecked) and an 'Order' of 100. A 'Filter Conditions' section is present with buttons for 'Add Filter Condition' and 'Add OR Clause'. A note at the top states: '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.' A 'More Info' link is also present.

5. Configure The Relationship:

The screenshot shows the ServiceNow interface for configuring a relationship. The left sidebar navigation includes 'Relationship' under 'FAVORITES'. The main content area is titled 'Relationship - Daily Expenses'. It displays the relationship's name ('Daily Expenses'), application ('Global'), and status ('Advanced' unchecked). The 'Applies to table' field is set to 'Family Expenses [u_st_u_auto...]' and the 'Queries from table' field is set to 'Daily Expenses [u_daily_expe...]'. A note at the top 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 is a code editor window containing ECMAScript 2021 (ES12) mode code:

```
1 (function refineQuery(current, parent) {  
2  
3  
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  
11 })(current, parent);  
12  
13
```

6. Conclusion

This project helped implement ServiceNow fundamentals such as custom application creation, table design, form layout, relationships, and update set tracking. It simulates a

Calculating Family Expenses Using ServiceNow...

real-world scenario of tracking family expenses and demonstrates the power of low-code/no-code platforms like ServiceNow.