M.A.M COLLEGE OF ENGINEERING AND TECHNOLOGY

Siruganur, Tiruchirappalli.

DEPARTMENT OF
INFORMATION TECHNOLOGY

A Project on

CALCULATING FAMILY EXPENSES USING SERVICE NOW

NM ID :

TEAM ID : NM2025TMID00751

TEAM SIZE : 4

GROUP MEMBER'S NAME WITH REG. NO :

Abisha A– 812022205001(Team Leader)

Ahamed Musthafa M– 812022205002 (Team Member)

Ahamed Yasik Sarvaththudeen M – 812022205003(Team Member)

Akash S– 812022205004(Team Member)

CERTIFICATE

This is to certify that the bonafide record of this work is done by Selvan/Selvi:…… Reg.No……. Of IV-Year VII- semester in ***Information Technology*** *For* ***NM1051 – ServiceNow Administrator Laboratory*** *during the academic Year 2025-2026.*

Faculty in Charge **HoD**

Submitted for the University Practical Examination held on……..

INTERNAL EXAMINER **EXTERNAL EXAMIN**

TABLE OF CONTENTS

IDEATION PHASE

Problem Statement

The project aims to develop a comprehensive expense calculation system using ServiceNow. This system will enable users to track and manage family expenses efficiently. It will include features such as expense categorization, budget setting, real-time tracking, and reporting capabilities. Utilizing ServiceNow's robust platform, the project will ensure seamless integration, user-friendly interface, and scalability to accommodate varying family sizes and financial complexities. The end goal is to empower users with the tools they need to make informed financial decisions and promote financial well-being within the family unit.

Problem Definition

In today's fast-paced world, families often face challenges in effectively managing their day-to-day expenses and maintaining financial stability. Traditional methods of expense tracking—such as manual record-keeping or using basic spreadsheets—are time-consuming, prone to errors, and lack real-time visibility into financial data. As a result, families struggle to monitor their spending habits, set realistic budgets, and make informed financial decisions.

Furthermore, with multiple family members contributing to expenses, it becomes difficult to consolidate and analyze overall spending patterns. The absence of a centralized system leads to poor financial planning, overspending, and limited control over household budgets. Many existing solutions are either too complex for non-technical users or lack the customization needed to suit different family structures and financial goals.

To address these issues, there is a need for an **automated, user- friendly, and integrated expense management system** that allows families to record, categorize, and track expenses efficiently. The proposed **Family Expense Management System**, developed on the **ServiceNow platform**, aims to fill this gap by providing real-time tracking, budget management, and reporting features. This solution will help families achieve better financial visibility, discipline, and long-term financial well-being.

Abstract

This project focuses on developing a **Family Expense Management System** using **ServiceNow** to help users efficiently track and manage household finances. The system will provide features such as **expense categorization**, **budget setting**, **real-time tracking**, and **comprehensive reporting**. By leveraging ServiceNow's robust and scalable platform, the solution ensures a **user-friendly interface**, **seamless integration**, and adaptability to various family structures. The ultimate goal is to empower families to make **informed financial decisions** and promote **financial well-being** through better expense visibility and control.

Leveraging the robust capabilities of the **ServiceNow platform**, the system will feature **automated workflows**, **budget tracking**, and **comprehensive reporting tools** to ensure

accuracy and convenience. It will also allow customization to accommodate varying family sizes and financial needs.

Empathy Map canvas

Section	Description
Users	Family members, parents, and individuals managing household budgets.
Says	“I need to know where my money goes every month.” “Tracking expenses manually takes too much time.”
Thinks	“If only I had an automated way to categorize and view my expenses.” “I want to save more efficiently.”
Does	Records expenses manually in spreadsheets or notebooks; sometimes forgets to log small expenses.
Feels	Overwhelmed by untracked spending, anxious about financial management, relieved when able to visualize spending clearly.

SAYS

- I need to know where my money goes every month.
- Tracking expenses manually takes too much time

THINKS

- 'If only I had an automated way to categorize and view my expenses.
- I want to save more efficiently

DOES

- Records expenses manually in spreadsheets or notebooks
- Sometimes forgets to log small expenses

FEELS

- Overwhelmed by untracked spending
- Anxious about financial management
- Relieved when able to visualize spending clearly

Users

Introduction

Managing family expenses effectively is essential for maintaining financial stability and achieving long-term financial goals. However, many families struggle to track daily spending, categorize expenses, and monitor budgets efficiently. To address these challenges, this project proposes the development of a **Family Expense Management System** built on the **ServiceNow platform**.

ServiceNow's powerful workflow automation, data management, and reporting capabilities make it an ideal platform for building a centralized expense tracking solution. The system will allow users to **record, categorize, and analyze expenses**, **set and monitor budgets**, and **generate insightful reports** for better financial planning. Designed with scalability and ease of use in mind, the solution can accommodate varying family sizes and financial complexities. Ultimately, this system aims to **simplify expense management**, **enhance financial transparency**, and **empower families to make informed financial decisions**.

The Family Expense Management System is designed to help families efficiently manage and monitor their financial activities using the ServiceNow platform. Managing household expenses can often become complex, especially when multiple members contribute to spending. This system provides a centralized solution where users can easily record, categorize, and track their daily expenses, ensuring greater visibility and control over family finances.

Built on ServiceNow, the project leverages the platform's workflow automation, data management, and reporting capabilities to deliver a smooth and efficient experience. Users can set monthly or yearly budgets, receive real-time updates on spending limits, and analyze financial data through interactive dashboards and reports. This not only helps in identifying unnecessary expenditures but also supports better financial planning and goal setting.

The system is designed to be user-friendly, scalable, and adaptable to families of varying sizes and financial complexities. It ensures that all financial data is securely stored and easily accessible, allowing users to make informed decisions based on accurate and up-to-date information. Through customization options, families can personalize categories and reports according to their unique financial needs.

Overall, the Family Expense Management System promotes financial awareness, discipline, and transparency within households. By simplifying expense tracking and budgeting through an intuitive digital platform, it empowers families to make smarter financial decisions and work towards long-term financial stability and well-being.

Objectives

Main objective of the Family Expense Management System is to develop a comprehensive and user-friendly platform on ServiceNow that enables families to efficiently manage their financial activities. To achieve this goal, the project focuses on the following specific objectives:

1. **Expense Tracking:**
   To provide an easy and efficient way for users to record, categorize, and manage daily household expenses in a centralized system.
2. **Budget Management:**
   To enable users to create, monitor, and adjust family budgets, ensuring spending remains within planned limits.
3. **Real-Time Monitoring:**
   To offer real-time visibility into financial data, allowing families to track their income and expenses instantly.
4. **Reporting and Analysis:**
   To generate detailed reports and visual dashboards that help users analyze spending patterns and identify areas for improvement.
5. **User-Friendly Interface:**
   To design an intuitive and accessible interface suitable for all family members, regardless of technical expertise.
6. **Data Security and Accuracy:**
   To ensure that all financial data is securely stored, processed, and presented accurately using ServiceNow's robust architecture.
7. **Scalability and Customization:**
   To develop a flexible system that can be customized and scaled to accommodate families of different sizes and financial complexities.
8. **Financial Awareness and Control:**
   To promote financial discipline, awareness, and informed decision-making within the family unit.

These objectives collectively aim to empower families with the tools they need to manage their finances effectively and achieve long-term financial well-being.

Scope

The Family Expense Management System aims to develop a comprehensive solution on the ServiceNow platform for managing household finances efficiently. The system will allow users to record, categorize, and monitor expenses, set and manage budgets, and generate detailed reports to analyze spending patterns. It will also feature real-time tracking, interactive dashboards, and customizable categories to meet the diverse financial needs of different families. Leveraging ServiceNow's capabilities, the system ensures a secure, scalable, and user-friendly experience for all users.

The project focuses on simplifying expense management and promoting financial awareness within the family unit. While the system will provide robust tools for expense tracking and budget control, it will not include features like direct bank integration, **automated payments,** or investment management. The primary scope is to empower families with a centralized platform that enhances financial visibility, discipline, and decision-making, leading to improved financial well-being

In-Scope Activities

- **Expense Recording and Categorization:**
- Allow users to input daily expenses and assign them to predefined or custom categories.
- Budget Creation and Management:
- Enable users to set monthly or yearly budgets for different expense categories and track adherence.
- Real-Time Expense Tracking:
- Provide instant updates on expenses and budget usage to give families a clear financial overview.
- Reporting and Analytics:
- Generate detailed reports and visual dashboards to analyze spending patterns and trends.
- User Interface Design:
- Develop a user-friendly and intuitive interface accessible to all family members, regardless of technical expertise

Out of Scope:

The following activities are **not included** in the scope of the **Family Expense Management System** project:

1. Bank Account Integration:
    - The system will not connect directly with banks or financial institutions to automatically fetch transactions.
2. Automated Bill Payments:
    - Scheduling or processing of payments for bills, loans, or subscriptions is not part of this project.
3. Investment and Asset Management:
    - Tracking investments, stocks, mutual funds, or other financial assets is excluded.
4. Credit/Debit Card Management:
    - The system will not handle card transactions, credit limits, or repayments.
5. Advanced Tax Calculations:
    - Calculation of taxes, tax filing support, or financial advisory services will not be provided.
6. Mobile Application Development:
    - The project will focus on the ServiceNow platform interface and will not include a separate mobile app

PERFORMANCE & TESTING

Procedure or Implementation steps

Phase 1 : Creation of New Update Set

1. Go to All >> In the filter search for Local Update set > click on New.
2. Enter the Details as:
   Name : Family Expenses

Phase 2 : Creation of New Update Set

1. Go to All >> In the filter search for Local Update set > click on New.
2. Enter the Details as:
   Name : Family Expenses
3. Then click on Submit and Make current.

Phase 3 : Creation of Table

Step 1: Creation of Family Expenses Table

1. Go to All > In the filter search for Tables > click on New.
2. Enter the Details:
   Label : Family Expenses
   Name : Auto-Populated

   New menu name : Family Expenditure
3. Go to the Header and right click there>> click on Save.

Step 2: Creation of Columns(Fields)

1. Near Columns Double click near insert a new row.
2. Give the details as

   Column label : Number

   Type : String

3. Double click on insert a new row again
4. Give the details as:

   Column label : Date

   Type : Date

5. Double click on insert a new row again
6. Give the details as:

   Column label : Amount

   Type : Integer

7. Double click on insert a new row again
8. Give the details as:

   Column label : Expense Details

   Type : String

   Max length : 800

Step 3: Configure the Form

1. Go to All >> In the filter search for Family Expenses >> Open Family Expenses
2. Click on New
3. Go to the Header and right click there>> click on Configure >> Select Form Design
4. Customize or Drag Drop the form as per your requirement.

Phase 4 : Creation of Relationship between Family Expenses and Daily Expenses tables

1. Click on all>> search for service catalog
2. Select maintain item under catalog definition
3. Search for ‘laptop request’ which is created before
4. Select ‘laptop request’ and scroll down click on “Catalog Ui policies”
5. In the catalog ui policies related list tab click on new
6. Give short description as: show accessories details

Phase 6 : Configuring Related List on Family Expenses

1. Go to All >> In the filter search for Family Expenses >> Open Family Expenses
2. Click on New
3. Go to the Header and right click there>> click on Configure >> Select Related Lists
4. Add Daily Expenses to the Selected Area.
5. Click on Save

Phase 7 : Creation of Business Rules

1. Go to All >> In the filter search for Business Rules.
2. Under System Definition Select Business Rules then click on New.

3. Enter the Details:
   Name : Family Expenses BR
   Table : Select Daily Expenses
   Check Advanced

Phase 8 : Configure the Relationship

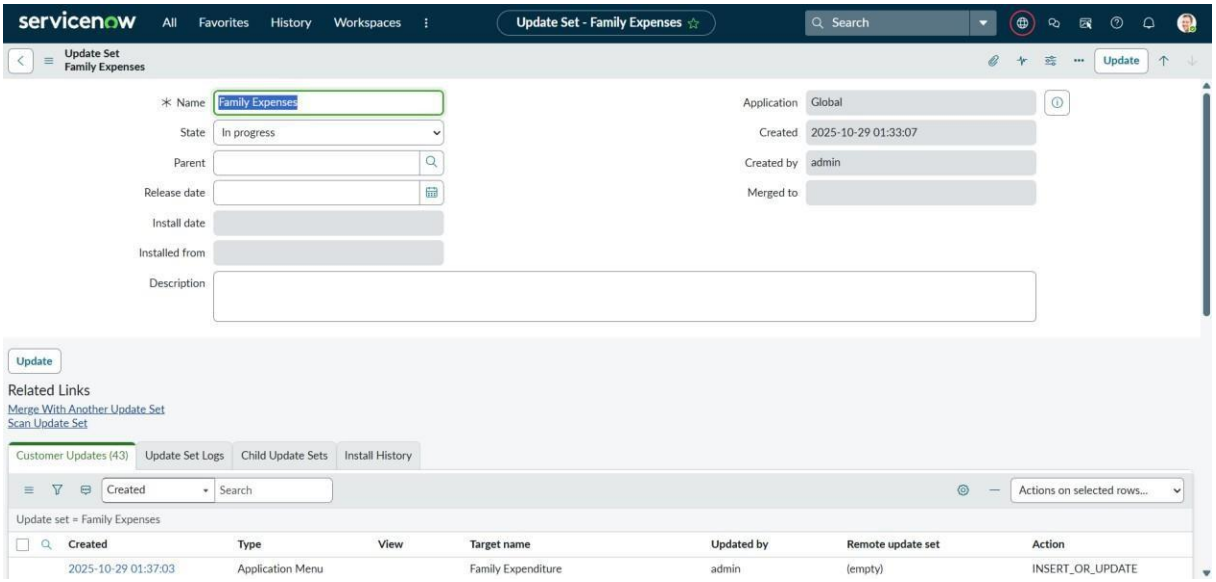
1. Go to All >> In the filter search for Relationships >> Open Relationships.
2. In that, open Daily Expenses Relationship.
3. For Applies to table : Select Family Expenses.
4. In Query with : write the below Query.

```
(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);
```

5. Click on Update.

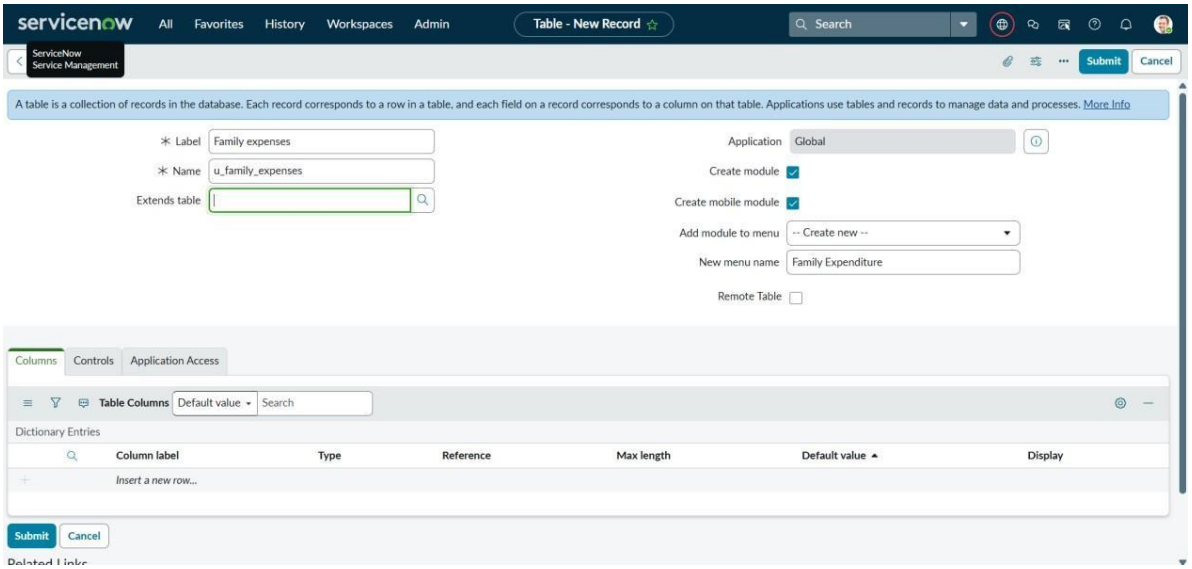
Screenshots:

Phase 1 : Creation of New Update Set

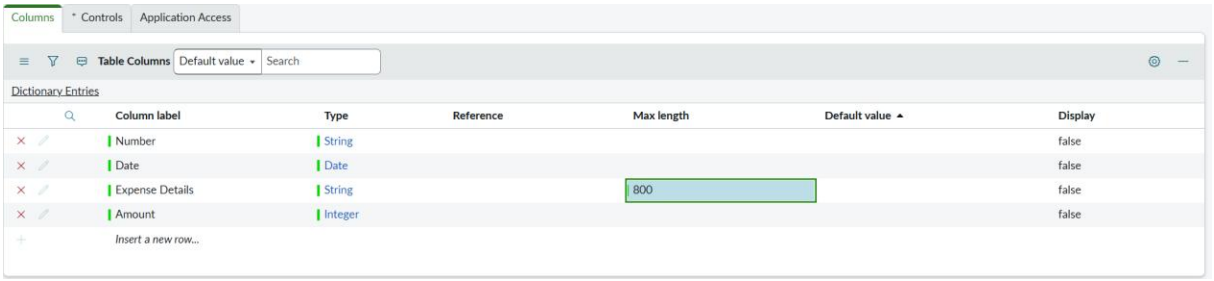


Phase 2 :

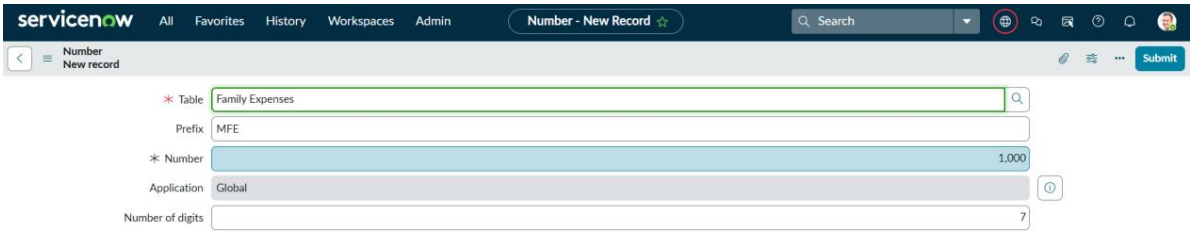
1. Creation of Family Expenses Table



2 . Create a columes

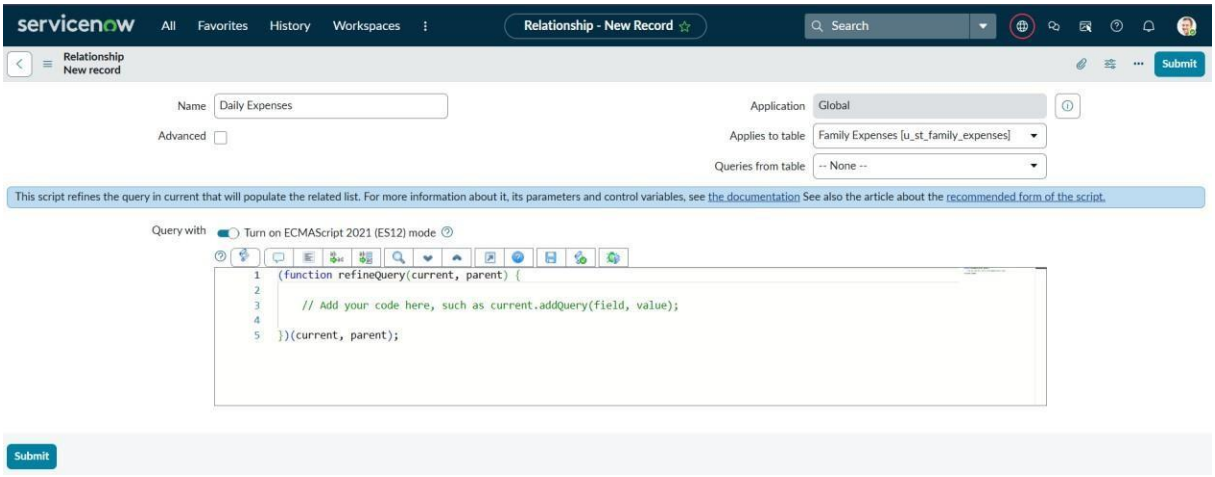


3 .Making Number Field an Auto-Number

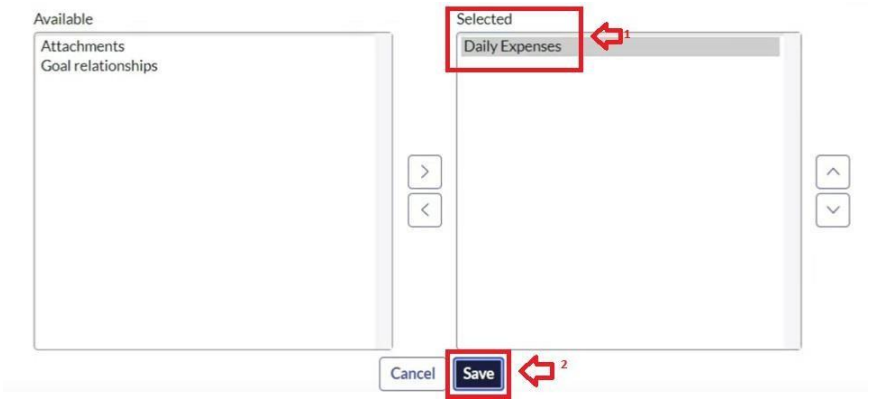


4.Configure the Form

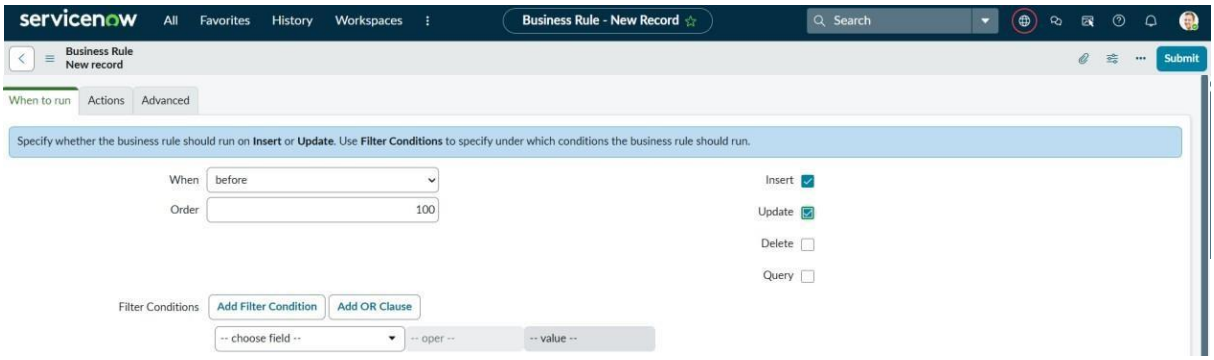
Phase 3 : Creation of Relationship between Family Expenses and Daily Expenses tables



Phase 4: Configuring Related List on Family Expenses



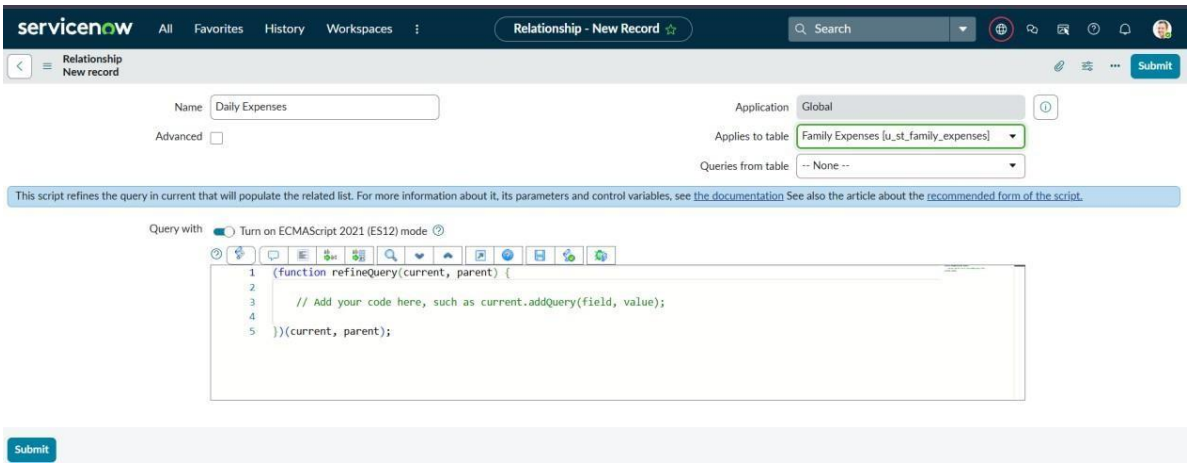
Phase 5: Creation of Business Rules



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);
```

Phase 6: Configure the Relationship


servicenow
All
Favorites
History
Workspaces
Relationship - New Record
Search
Relationship
New record
Submit
Name
Daily Expenses
Advanced
Application
Global
Applies to table
Family Expenses [u_st_family_expenses]
Queries from table
-- None --
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.
Query with
Turn on ECMAScript 2021 (ES12) mode
(function refineQuery(current, parent) {
// Add your code here, such as current.addQuery(field, value);
})(current, parent);
Submit

PROJECT DESIGN PHASE

Problem – Solution fit

Managing and tracking family expenses manually is time-consuming and prone to errors. Families often struggle with categorizing expenses, setting budgets, and analyzing financial patterns, leading to poor financial decisions and stress.

Proposed Solution

Develop a **ServiceNow-based Expense Management System** that automates expense tracking, categorization, and analysis. The system will allow users to:

- Add expenses under customizable categories (e.g., food, rent, transport).
- Set monthly or yearly budgets.
- Receive real-time updates and graphical expense reports.
- Enable multiple family members to contribute data collaboratively. This ensures efficient financial tracking, transparency, and improved financial decision-making.

Parameter	**Description**
Platform	ServiceNow (leveraging its workflow automation and UI design).
Objective	To automate and streamline family expense tracking and budgeting.
Users	Family members, individuals, or small groups managing shared finances.
Features	Expense categorization, budgeting, expense reports, and family sharing.
Outcome	Efficient financial management and data-driven decision-making.
Technology Stack	ServiceNow, JavaScript, GlideRecord API, and ServiceNow UI Builder

Conclusion

The **Family Expense Management System using ServiceNow** provides an efficient and user-friendly solution for managing and tracking household expenses. By incorporating features like expense categorization, budgeting, real-time tracking, and reporting, the system empowers families to make informed financial decisions and maintain better control over their spending. Leveraging ServiceNow's scalability and automation capabilities, the project ensures smooth integration, reliability, and adaptability for various financial needs. Overall, it promotes financial discipline, transparency, and well-being within the family, making it a valuable tool for effective personal finance management.

Solution Architecture

1. Goals of the Architecture

- Provide seamless integration within ServiceNow's platform.
- Enable real-time expense tracking and categorization.
- Support user-friendly dashboards and automated workflows.
- Ensure scalability for various family sizes and expense complexities.

2. Key Components

- **User Interface (UI):** Built on ServiceNow's UI Builder for intuitive navigation.
- **Expense Entry Module:** Allows users to input and categorize expenses.
- **Budget Manager:** Enables users to set spending limits and track them dynamically.
- **Analytics & Reporting:** Generates visual reports and insights for better decisions.
- **Database (ServiceNow Tables):** Stores user data, categories, and expense details.
- **Notification Engine:** Alerts users for budget limits or unusual spending.

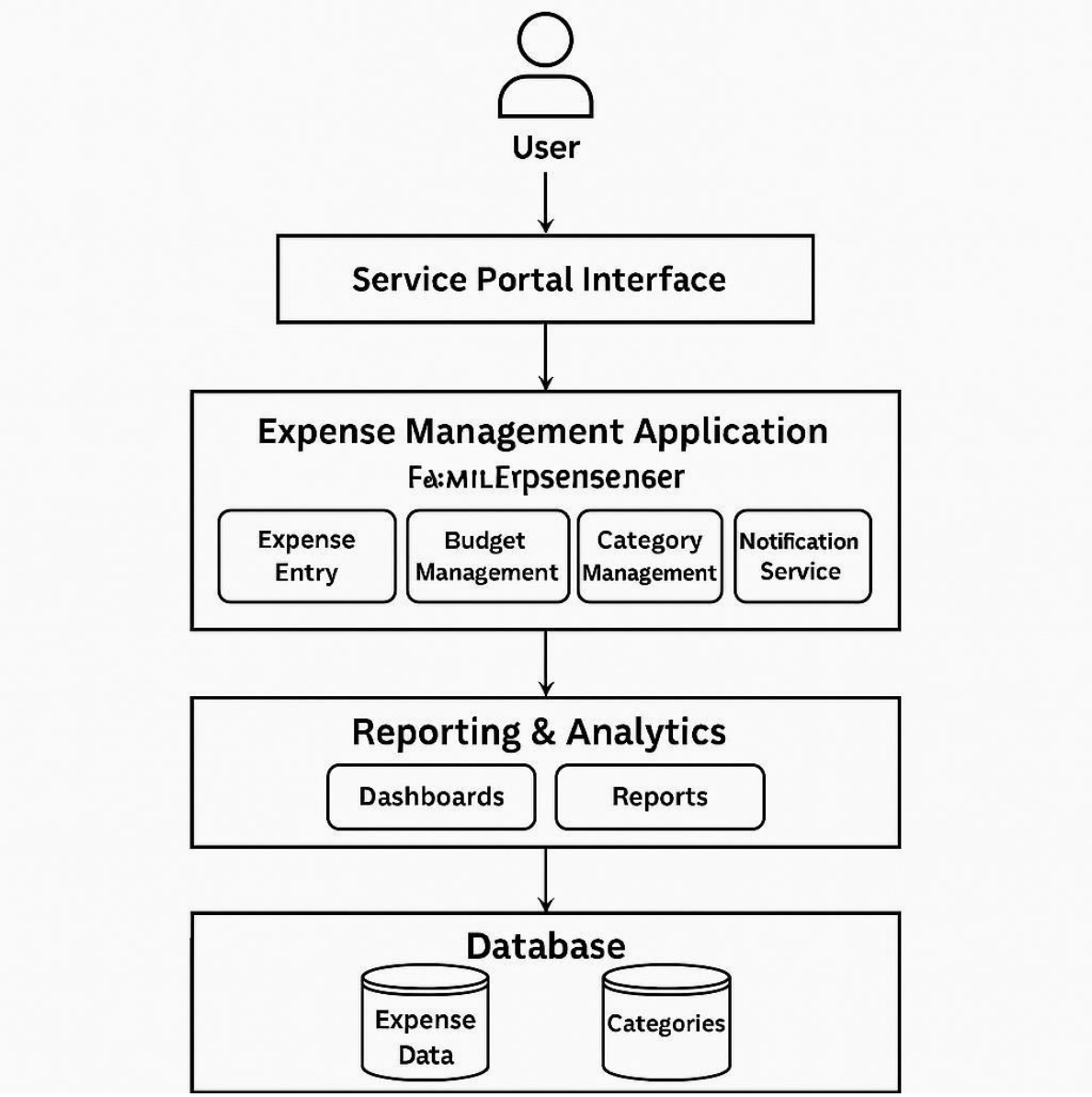
3. Development Phase

- Requirement Analysis: Identify routing rules, user roles, and automation needs.
- System Design: Plan architecture, workflows, and data flow structure.
- Development: Configure modules, scripts, and automated flows in ServiceNow.
- Testing & Validation: Verify routing accuracy, notifications, and SLA tracking.
- Deployment: Move configurations to the production environment.
- Monitoring & Optimization: Continuously analyze performance and refine rules.

Solution Architecture Description

The **Solution Architecture** for the Family Expense Management System is designed to create an efficient, secure, and scalable platform that helps users track, manage, and analyze their household expenses. Built on the **ServiceNow** platform, the architecture integrates multiple components — user interface, database, workflow automation, and analytics — to deliver a seamless financial tracking experience.

Solution Architecture Diagram



PROJECT PLANNING PHASE

Project Planning Phase

1. Product Backlog

ID	Feature	Description
PB1	Expense Entry Module	Allows users to add, edit, and delete expenses.
PB2	Budget Management	Enables users to set monthly and yearly budgets.
PB3	Expense Categorization	Automatically classifies expenses into defined categories.
PB4	Reporting & Dashboard	Provides visual insights and expense summaries.
PB5	Notification Service	Sends alerts when spending exceeds the budget.
PB6	Data Export	Allows users to download reports in CSV or PDF format.
PB7	Security & Access Control	Ensures user authentication and data protection.

2. Sprint Planning

Sprint	Duration	Major Tasks	Deliverables
Sprint 1	2 Weeks	Requirement gathering, UI mockups, database design	Requirement document, database schema
Sprint 2	2 Weeks	Develop Expense Entry & Categorization modules	Working expense entry and category UI
Sprint 3	2 Weeks	Implement Budget Management and Notification service	Budget alerts, limit tracking
Sprint 4	2 Weeks	Add Reporting & Dashboard features	Reports and charts
Sprint 5	1 Week	Testing, debugging, and deployment	Final deployed system

3. User Stories

User Story ID	As a (User)	I want to...	So that I can...
US1	Family Member	Record daily expenses	Track and manage spending
US2	Family Head	Set monthly budgets	Control overspending
US3	User	Get categorized reports	Understand spending patterns
US4	User	Receive budget alerts	Stay informed about financial limits
US5	Admin	Manage users and access rights	Ensure data privacy and control

4. Story Points

User Story ID	Story Points	Complexity Level
US1	5	Medium
US2	8	High
US3	6	Medium
US4	4	Low
US5	7	High

REQUIREMENT ANALYSIS

1. Solution Requirements

Requirement Type	Parameter	Description
Functional Requirement	Expense Entry Module	Users can add, edit, and delete expenses under various categories.
Functional Requirement	Budget Management	Allows users to define and track monthly or annual budgets.
Functional Requirement	Expense Categorization	Automatically classifies expenses based on predefined tags.
Functional Requirement	Reporting & Dashboard	Provides real-time visual reports and analytics on expenses.
Functional Requirement	Notification System	Sends alerts for budget limits, due payments, or unusual expenses.
Functional Requirement	Multi-user Access	Supports multiple family members with role-based permissions.
Non-Functional Requirement	Performance	System must handle multiple concurrent users efficiently.
Non-Functional Requirement	Security	Implements authentication, data encryption, and role-based access control.
Non-Functional Requirement	Scalability	Architecture should support expanding data and user base without performance loss.
Non-Functional Requirement	Usability	The interface should be intuitive and accessible to non-technical users.
Non-Functional Requirement	Reliability	System must ensure accurate data storage and quick recovery in case of failure.
Technical Requirement	Platform	Built on ServiceNow using Flow Designer, Business Rules, and Service Portal.
Technical Requirement	Database	Uses ServiceNow Tables for structured data storage.
Technical Requirement	Integration	Allows external data import/export via REST APIs.

2. Data Flow Diagram (DFD – Level 1)

Description:

The Expense Calculation System DFD shows how data moves through the system to manage and track family expenses efficiently.

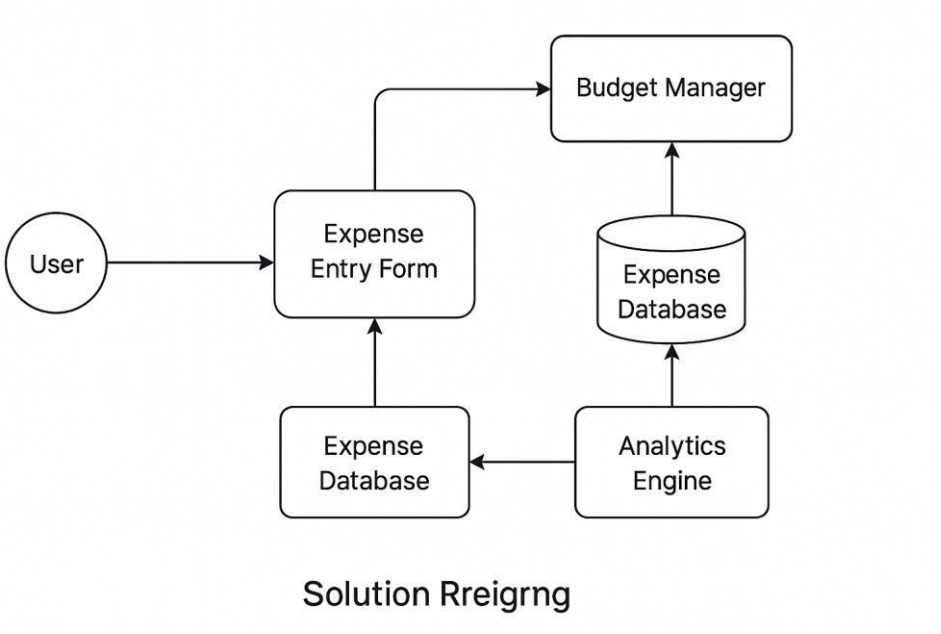
- **Users input expenses, set budgets, and view reports.**
- **Admin manages user accounts and system data.**

Main Processes:

1. **Expense Input – Users add expense details, which are validated and stored.**
2. **Budget Setting – Users define budgets; the system monitors and triggers alerts if limits are exceeded.**
3. **Categorization – Expenses are automatically grouped (e.g., food, bills, travel).**
4. **Report Generation – The system creates summaries and visual reports.**
5. **Notifications – Alerts users about budget status and spending trends.**

Data Stores:

- **User Database**
- **Expense Data**
- **Budget Records**
- **Report Repository**



3. Technology Stack

Layer / Category	Technology / Tool	Description / Purpose
Platform	**ServiceNow**	Core platform used for building, hosting, and managing the expense management application.
Frontend (User Interface)	**Service Portal / Now Experience UI Builder**	Provides a user-friendly interface for expense input, reports, and dashboards.
Backend (Logic Layer)	**Flow Designer, Business Rules, Script Includes**	Handles data processing, automation, and workflow logic for expense calculations.
Database	**ServiceNow Tables**	Stores user details, expense data, budgets, and reports securely.
Integration	**REST API / Import Sets**	Enables data import/export from external sources like bank statements or spreadsheets.
Reporting & Analytics	**Performance Analytics / Reporting Module**	Generates visual dashboards and analytical reports for tracking expenses.
Security	**Role-Based Access Control (RBAC)**	Ensures authorized access and data protection for all users.
Notifications	**Email Notifications / Flow Actions**	Sends budget alerts, reminders, and summaries to users automatically.
Development Tools	**ServiceNow Studio, JavaScript, Glide API**	Used to design and develop custom logic and automation within the platform.
Deployment & Governance	**Update Sets / Application Repository**	Manages deployment, version control, and governance across environments.

GENERATIVE AI IN ACTION

Artificial intelligence has evolved through numerous phases, yet it remains both intriguing and inspiring to witness machines become increasingly capable of crafting poetry, humor, and responses that uncannily mimic human creativity. You'll learn about the history of AI, how deep learning plays a pivotal role in generative AI ("gen AI"), and how gen-AI works and is applied to various industries. Throughout this course, you will also learn how to create algorithms, and gain hands-on experience writing code using popular programming languages.

Completed the following required modules to earn an industry-recognized IBM SkillsBuild digital credential called **Generative AI in Action:**

1. Introduction to Generative AI
2. Crafting Precision Prompts with Generative AI
3. Coding Simplified with Generative AI

After completing Generative AI in Action, I was able to:

- Explain how generative artificial intelligence works
- Define what foundation models are and their role in machine learning
- Understand how transformers models are used to solve various language-related tasks
- Describe how prompt engineering improves generative AI models
- Identify common prompt elements
- Explore code generation using a natural language prompt
- Describe prompt engineering techniques
- Perform common programming tasks using Python's built-in functions and libraries
- Create scripts and code for solving real-world problems and automating routine tasks

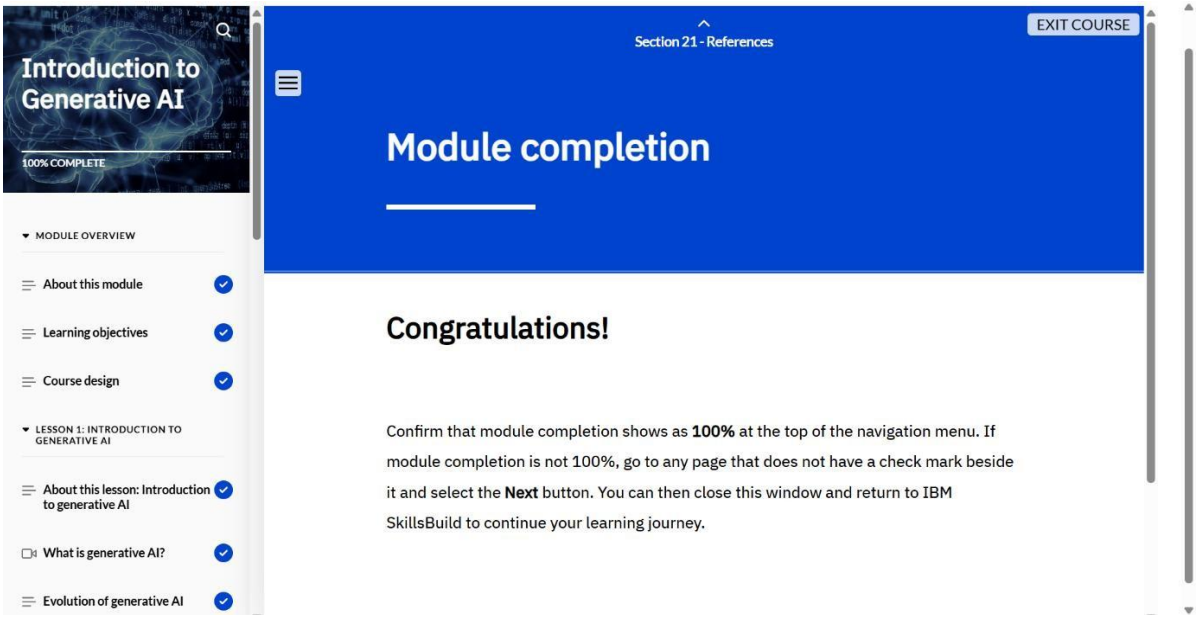
MODULE 1 - Introduction to Generative AI

In this course, I have learned about the history of AI, how deep learning plays a pivotal role in generative AI (gen-AI), and how gen-AI works and is applied to various industries. I also learned how to create algorithms, and gain hands-on experience writing code using popular programming languages.

After completing this module, I was able to:

- Explain how generative artificial intelligence (gen-AI) works
- Define what foundation models are and their role in machine learning

- Understand how transformers models are used to solve various language-related tasks
- Describe how prompt engineering improve generative AI models
- Perform common programming tasks using Python's built-in functions and libraries
- Create scripts and code for solving real-world problems and automating routine tasks



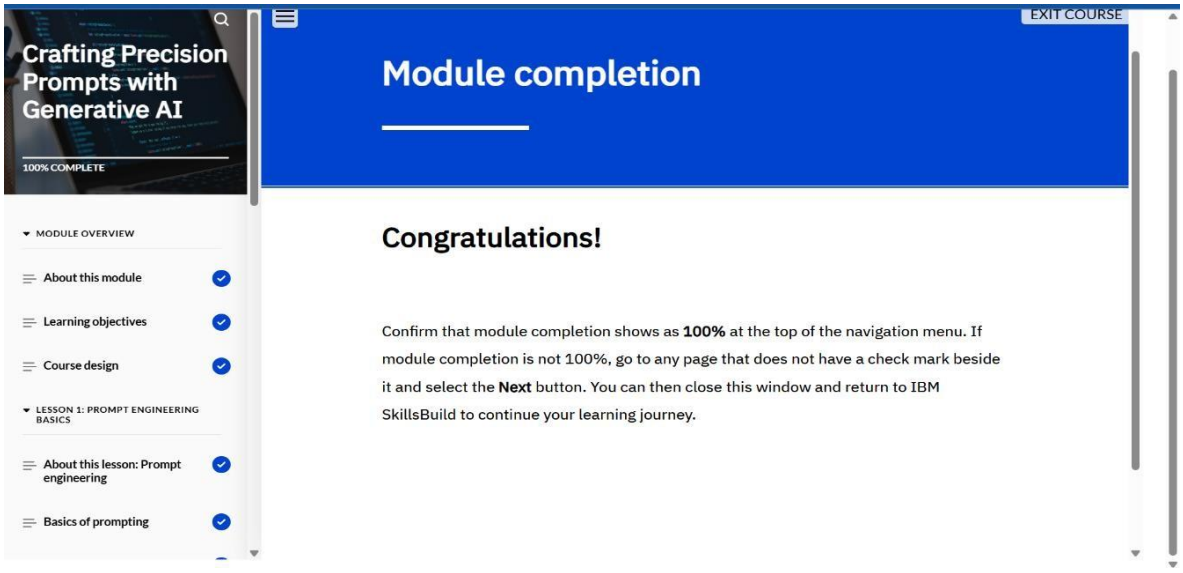
MODULE 2- Crafting Precision Prompts with Generative AI

This was an activity-based course. I learned about AI language models and the rules to follow when giving instructions, or prompting, an AI language model. I walked-through a guided activity that demonstrates how to write effective prompts for an AI language model to help plan a travel itinerary. Finally, I participated in an activity to apply what I learned to effectively write prompts for an AI language model to create my own custom music playlist.

After completing this course, I was able to:

- Describe an AI language model
- Explain how an AI language model understands and responds to humans
- Identify the rules to follow to write effective prompts to generate focused and accurate results from an AI language model
- List the steps to sign up for a ChatGPT account
- Follow the steps to effectively write and refine a series of prompts for ChatGPT for a travel itinerary scenario

- Demonstrate the steps to effectively write and refine a series of prompts for ChatGPT to create a custom music playlist

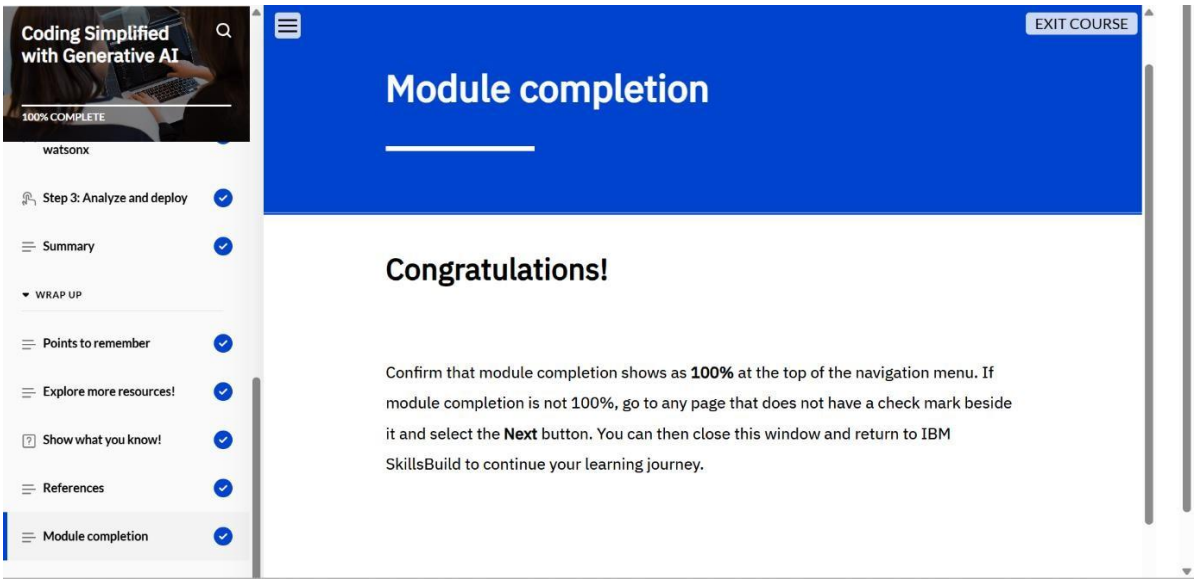


MODULE 3- Coding Simplified with Generative AI

In this course, I had learned the basics of scripting, understand its distinctions from traditional programming, and explore how generative AI models are used to simplify and streamline code generation. Through hands-on labs, I also learned how to create algorithms and apply my skills using widely used programming languages.

After completing this module, I was able to:

- Define scripting and how it is works
- Explain the differences between scripting and traditional programming and when each approach is used
- Describe how Python is used to perform various tasks
- Create a working Python application using IBM watsonx Code Generation



CONCLUSION

The Generative AI in Action program provided a solid foundation in understanding the core principles and practical applications of generative artificial intelligence. Through the three modules—Introduction to Generative AI, Crafting Precision Prompts with Generative AI, and Coding Simplified with Generative AI—I developed a deep understanding of how AI models such as foundation and transformer models function, and how prompt engineering enhances their effectiveness. The hands-on activities improved my ability to craft accurate prompts, automate coding tasks, and generate creative as well as technical outputs using Python and AI-assisted tools. Overall, this program strengthened my knowledge of AI-driven innovation, improved my coding proficiency, and equipped me with essential skills to apply generative AI techniques in real-world scenarios across multiple industries.

WELCOME TO SERVICE NOW

Practice Scenarios for ServiceNow Admin

1. **Create a new user for a contractor, assign them to an "IT Support" group, and ensure they can only access the *Incident* application.**
Solution:

- **Create the Contractor User**
- Navigate to **Users** → *User Administration > Users*.
- Click **New**.
- Fill in details:
    - **User ID**: contractor1
    - **First name / Last name**: Contractor User
    - **Email**: contractor1@gmail.com
    - **Active**: Checked.
- Save.
- **Assign the User to the "IT Support" Group**
- On the user record, scroll to **Groups** (related list).
- Click **Edit**.
- Add to the **IT Support** group.
- Save.
- **Restrict Access to Only the Incident Application**

Now we need to make sure this contractor can only work with **Incident**.

Option A: Role-Based Control (Mostly Preferred)

- By default, Incident application requires **itil** role.
- Instead of giving full **itil** access (which gives too much), do the following:
    - Create a **new custom role**, ex: **incident_contractor**.
    - Assign this role only to permissions needed for Incident (using ACLs).
    - Assign the new role to your contractor user.
    - Do **not** give **itil** or other broad roles.

Option B: Application Menu Restriction

- Go to **System Definition > Application Menus**.
- Open the **Incident** application menu.
- In the **Roles** field, add your custom role (**incident_contractor).**
    - This ensures only users with this role can see the Incident.
- **Verify Access**
- **Impersonate** the contractor user.
- Check:
    - They should only see the **Incident application** in the left nav.

- They can open/create/edit incidents (based on the ACLs you configured).
- They cannot access other apps (like Change, Problem, etc.).

2. **Assign a role to a new group so members can read *Knowledge Articles* but cannot create or edit them.**

- ***Create a New Group***
- Navigate to User Administration > Groups.
- Click New.
- Enter a Name for the group (e.g., Knowledge Readers).
- Optionally, add a Description.
- Click Submit.
- ***Assign the Appropriate Role***

To allow read-only access to Knowledge Base articles, assign the **knowledge** role:

- Open the newly created group.
- Scroll to the Roles related list.
- Click Edit.
- Add the role: knowledge
  - This role allows users to view published articles.
- Click Save.

Do NOT assign roles like **knowledge_admin or knowledge_manager, which grant create/edit permissions.

- ***Add Users to the Group***
- In the group record, scroll to the Group Members related list.
- Click Edit.
- Select users you want to add.
- Click Save.
- ***Verify Access***
- Log in as one of the group members.
- Navigate to Knowledge > Articles.
- Confirm they can view articles.
- Try creating or editing an article — they should not have access.

3. **Configure a UI Policy that hides the "Work Notes" field unless the state is "In Progress"**. **Solution:**

- **Navigate to UI Policies**
- Go to Application Navigator → type UI Policies → click System UI > UI Policies.
- Create a New UI Policy
- Click New.
- Select the Table → e.g., *Incident* (or whichever table you're working on).
- Provide a Name (e.g., *Hide Work Notes unless In Progress*).
- In the Conditions section, set:

    - Field = *State*
    - Operator = *is*
    - Value = *In Progress*.
- Check the box Active.
- Save the record.
- **Add a UI Policy Action**
- In the same UI Policy record, scroll to UI Policy Actions (Related List).
- Click New.
- Configure the action:
    - Field name = *Work notes*
    - Visible = *True* (since you want it visible only when the condition is met).
- Submit the action

4. **Configure a UI Policy to hide Notes section in incident, when state is In Progress.**

Solution:

- **Navigate to UI Policies**
- Go to Application Navigator → type UI Policies → click System UI > UI Policies.
- Create a New UI Policy
- Click New.
- Select the Table → e.g., *Incident* (or whichever table you're working on).
- Provide a Name (e.g., *Hide Work Notes unless In Progress*).
- In the Conditions section, set:
    - Field = *State*
    - Operator = *is*
    - Value = *In Progress*.
- Check the box Active.
- Save the record.
- **Make Run Script box True**
- Just write one line of code:
- g_form.setSectionDisplay('notes',false);
- Submit the action

5. **Configure a response SLA, the SLA should pause, when the incident state is in On Hold vice versa**.

- ***Create or Modify an SLA Definition***
- Navigate to **Service Level Management > SLA Definitions**.
- Click **New** or open an existing SLA (e.g., "Response SLA").
- Fill in the basic details:
    - **Name**: Response SLA
    - **Table**: **Incident**
    - **Type**: Response
    - **Duration**: Set your desired time (e.g., 1 hour)
- ***Set SLA Conditions***

- Under the **Start Condition**:
    - Example: **State is New**
- Under the **Stop Condition**:
    - Example: State is Resolved or Closed
- Under the **Pause Condition**:
    - Add: **State is On Hold**

This ensures the SLA timer **pauses** when the incident is moved to **On Hold**, and **resumes** when it returns to another **New** state

- ***Test the SLA Behavior***
- Create a test incident.
- Confirm SLA starts when an incident is created.
- Change state to **On Hold** — SLA should pause.
- Change back to **Active** — SLA should resume.
- Resolve the incident — SLA should stop.

6. **Configure an email notification that alerts the assigned group whenever a new *Change Request* is created.**

Solution:

- **Navigate to Notifications**
- In the **Application Navigator**, type **Notifications**.
- Go to **System Notification > Email > Notifications**.
- **Create a New Notification**

1. Click **New**.
2. Fill in the basic details:
    a. **Name**: *New Change Request Assigned Group Alert*
    b. **Table**: *Change Request [change_request]*
    c. **Active**: Checked

- **Define When to Send**

1. Under **When to send**, configure:
    a. **When to send**: *Insert* (since you want this when a new record is created).

- **Define Who Will Receive**

1. In the **Recipients** tab:
    a. Under **Users/Groups in fields**, choose **Assigned to group** (or the field name for assigned group).
    b. This ensures the entire assigned group gets the email.

- **Define What Will Contain**
- In the **What it will contain** tab:
    - **Subject**: New Change Request Created - ${number}

      Message HTML (sample):

      A new Change Request has been created.

- Number: ${number}
- Short Description: ${short_description}
- Requested By: ${requested_by}
- Assignment Group: ${assignment_group}
- State: ${state}

Please review and take necessary action.

- **Save & Test**
- Save the Notification.
- Create a new **Change Request** record, assign it to a group.
- Verify that the email goes out to all members of the Assigned Group.

7. **Create a report showing the number of incidents opened by each department in the last 30 days.**

- ***Navigate to Reports***
- Go to Reports > Open Reports Modules.
- Click Create a Report.
- ***Define Report Source***
- Name: **Incidents by Department - Last 30 Days**
- Source Table: **Incident**
- ***Set Conditions***
- **Under Filter, add:**
  - Opened At → on or after → Today - 30 days
  - Department → is not empty ***(optional, to exclude unassigned)***
- ***Choose Report Type***
- Select Type: **Bar Chart** or **Pie Chart** (or **List** if you prefer tabular view)
- ***Configure Grouping***
- Under Group By, select: **Department**
- Under Aggregation, choose: **Count**
- ***Save and Run***
- Click Save.
- Click Run to view the report.

8. **Build a dashboard for Service Desk Managers showing KPIs like incidents by priority, created within a week, state wise also.**

Step 1: Create Individual Reports

You'll need to create three separate reports first:

- ***Incidents by Priority***
- Go to: Reports > Create New
- Name: Incidents by Priority
- Source Table: Incident

- Type: Bar Chart or Pie Chart
- Group By: Priority
- Aggregation: Count
- Filter: Opened At → on or after → Today - 30 days
- ***Incidents Created Within a Week***
- Name: Incidents Created - Last 7 Days
- Source Table: Incident
- Type: Time Series or Bar Chart
- Filter: Opened At → on or after → Today - 7 days
- Group By: Opened At (Daily)
- Aggregation: Count
- ***Incidents by State***
- Name: Incidents by State
- Source Table: Incident
- Type: Bar Chart or Pie Chart
- Group By: State
- Aggregation: Count
- Filter: Opened At → on or after → Today - 30 days

Step 2: Create a Dashboard

- Go to Self-Service > Dashboards.
- Click Create New Dashboard.
- Name: **Service Desk Manager KPIs**
- Add a Proper Description
- Click Submit.

Step 3: Add Reports to the Dashboard

1. Open the newly created dashboard.
2. Click Edit Content.
3. Use Add Reports to include:
    - **Incidents by Priority**
    - **Incidents Created - Last 7 Days**
    - **Incidents by State**
4. Arrange the widgets as needed for clarity.

9. **Restrict the ability to delete records in the *Change Request* table so only users with the "admin" role can do so.**

- **Navigate to Access Control (ACLs)**
- In the **Application Navigator**, type **Access Control**.
- Go to **System Security > Access Control (ACL)**.
- **Create a New ACL Rule**
- Click **New**.
- Fill in details:

    - **Type**: *record*
    - **Operation**: *delete*
    - **Table**: *Change Request [change_request]*
    - **Name**: *(auto-populates when you pick table + operation)*
- **Define the Condition / Role**

  In the **Requires role** field, add: **admin**

  - This ensures only users with the **admin** role can delete records.
- **Save & Test**
- Save the ACL.
- Test with a non-admin user → they should **not** see the delete option (or get a permission error if they try via URL).
- Test with an admin user → delete should work normally.

10. **Create a custom table and create two reference fields (ex: assignment group and assigned to). Display the users based on selection of assignment group.**

- **Create a Custom Table**

1. In the Application Navigator, type **Tables**.
2. Go to **System Definition > Tables**.
3. Click **New**.
    - Name: *u_custom_task*
    - Label: *Custom Task.*
    - Save.

- **Add Fields**

1. Open your table and go to the **Columns** tab.
2. Add two reference fields:
    - **Assignment Group** → Type = *Reference*, Table = *sys_user_group*.
    - **Assigned To** → Type = *Reference*, Table = *sys_user*.

- **Configure Reference Qualifier on "Assigned To"**
- We need to filter "Assigned To" users based on the selected Assignment Group.

  Using Reference Qualifier

- Right click on the **Assigned To** field, click on **Configure Dictionary.**
- Go to **Dependent** Section, give the name of the Assignment Group(ex: u_ass_group)
- Update and Test the functionality.

11. **How to auto assign incidents when user selects a category as network, the same incident be assigned to Network group.**

Solution:

1. Go to Flow Designer → Designer.
2. Click New Flow.

- Name: Assign Incident by Category
- Trigger: Created or Updated → Table = Incident

3. Add a If action (Condition) with expression:
   - Select Trigger Record Category is Network
4. Under the If branch, add Action → Update Record:
   - Record: Trigger → Incident(Trigger Record)
   - Set field Assignment group → Network
5. Save and Activate the flow.
6. Test the Flow.

12. HR Groups members are only able to see HR Related Records in servicenow?

Solution:

Step 1: Create a Role for HR Access

Navigate to:
User Administration → Roles → New

1. Enter:
   - Name: hr_access
   - Description: Role to allow access to HR Cases
2. Click Submit.

Step 2: Assign the Role to HR Group

1. Navigate to:
   User Administration → Groups
2. Open your HR group record.
3. In the Roles tab → click Edit.
4. Move hr_access from Available → Selected.
5. Click Save.

Now all members of the HR group have the hr_access role.

Step 3: Create Access Control (ACL) for Viewing HR Cases

1. Navigate to:
   System Security → Access Control (ACL)
2. Click New.

Fill in:

Field	**Value**

Type	record
Operation	read
Table	Your HR Case table
Active	True

Step 4: Define Access Condition (No Script)

Scroll down to the Requires role section:

- Add the Role hr_access.

This means only users with the hr_access role can read/view HR Case records.

Step 5: Save and Test

1. Click Submit or Update to save the ACL.
2. Impersonate a non-HR user:
    - Go to your profile → click Impersonate User → choose a user *not in the HR group*.
    - Try opening an HR Case record → You should see a "Security constraints prevent access to requested page" message.
3. Now impersonate an HR group member:
    - They should be able to open HR Cases normally

13. When the Incident state changes to In Progress, Child incident related list should be hidden.

Solution:

1. Navigate to System UI → UI Policies → New.
2. Fill the header:
    - Name: Hide related lists when State is In Progress
    - Table: Incident
    - Active: checked
    - Global: checked
3. Condition: **State is In Progress**
   (Use the exact label used in your instance for the In Progress state.)
4. Submit the UI Policy record.

5. In the UI Policy record click **New** under **UI Policy Actions**.

Set:

- **Field name:** select the related list–Child incident
- **Visible:** false
- **Read only:** optional
- Save and Test the UI Policy Action.

14. How to Display Incident number while loading the incident form

Solution:

1. Navigate to System UI → Client Scripts → New.
2. Fill the header:
    - Name: Show Incident Number on Load
    - Table: Incident
    - Type: onLoad
    - Active: True
3. Add this script:

```
function onLoad() {

  // Get the Incident number field value

  var incNum = g_form.getValue('number'); // 'number' is the field name

  alert('Incident Number: ' + incNum);

}
```

15. When the Incident state changes to In Progress, description should be hidden and short description should be mandatory.

Solution:

Step 1 — Navigate to Client Scripts

1. Go to:
   System UI → Client Scripts → New
2. Fill the header:
    - Name: Hide Description and Make Short Description Mandatory
    - Table: Incident
    - Type: onChange
    - Field name: state
    - Active: checked

Step 2 — Add the Client Script Code

```
function onChange(control, oldValue, newValue, isLoading)
   { if (isLoading) return;
   if (newValue === '2')
      { g_form.setDisplay('description', false);
      g_form.setMandatory('short_description', true);
   } else {
       g_form.setDisplay('description', true);
      g_form.setMandatory('short_description', false);
   }
}
```

- Click **Submit** or **Update** to save.

15. If the description field is empty in the incident table, prevent the form submission.

Solution:

Step 1 — Navigate to Client Scripts

1. Go to:
   System UI → Client Scripts → New
2. Fill the header:
   - Name: Prevent Submit if Description Empty
   - Table: Incident
   - Type: onSubmit
   - Active: checked

Step 2 — Add the Client Script Code

```
function onSubmit() {
 var description = g_form.getValue('description');
   if (description == '') {
  g_form.addErrorMessage('Description cannot be empty');
      return false;
   } else
   { return
   true;
}
```

}

16. Users can not change the state field values in the incident list.

Solution:

Step 1 — Navigate to Client Scripts

3. Go to:
   System UI → Client Scripts → New
4. Fill the header:
   - Name: Prevent State Inline Edit
   - Table: Incident
   - Type: onCellEdit
   - Field name: state
   - Active: checked

Step 2 — Add the Client Script Code

```
if(newValue==2){

alert('You can not edit this value');

saveAndClose==false;

}else{ saveAndClose==tru

e;

}
```

17. How to set the Caller to Logged in user automatically in the incident table.

Solution:

1. Navigate: System Definition → Business Rules → New
2. Fill the details:
   - Name: Set Caller on Incident Create
   - Table: Incident
   - When: before
   - Insert/update: checked
   - Advanced: true
3. **Script**:

```
current.caller_id = gs.getUserID();
```

18. When a user updates an incident record, priority should change to Critical automatically.

Solution:

1. Navigate: System Definition → Business Rules → New

2. Settings:
    - Name: Set Priority field
    - Table: Incident
    - When: before
    - Update:checked
3. Script:

```
current.impact = 1;

current.urgency = 1;
```

19. Create a button on the Incident form that allows users to mark an Incident as Resolved with a single click.

Solution:

1. Navigate: System UI → UI Actions → New
2. Settings:
    - Name: Resolve Incident
    - Table: Incident
    - Action type: Form button
    - Active: checked
3. Script:
    - current.state = 6;
    - current.update();
    - action.setRedirectURL(current);

20. Create a button on the incident table that copies the Short Description value into the Description field.

Solution:

1. Navigate: System UI → UI Actions → New
2. Settings:
    - Name: Copy Short Description
    - Table: Incident
    - Action type: Form button
    - Active: checked

3. Script:
    - current.description = current.short_description;
    - current.update();
      action.setRedirectURL(current);