

Geolocation Data Management ServiceNow

Project Documentation

1. Introduction

- **Project Name:** Geolocation Data Management

2. Project Overview

- **Objective:**
To demonstrate proficiency in **ServiceNow full-stack development** by creating an application that integrates a Service Portal UI with a custom table and an external REST API, automating a key business process.
- **Description:**
Developed an end-to-end ServiceNow application for **Geolocation Data Management**. The solution features a modern Service Portal widget that captures user input and, through a robust **server-side script**, initiates a secure **REST API call** to ipgeolocation.io. The application dynamically parses the complex JSON response and persists the data in a custom table using Glide Record, showcasing expertise in **data modelling and external system integration**.

3. Project Ideation Phase

- **Project Title:** Geolocation Data Management
- **Problem Statement:** To automate the process of retrieving, storing, and displaying geolocation information for any given IP address, eliminating the need for manual lookups and providing a centralized data repository for analysis and reporting.

4. Requirement Analysis Phase

- **Tables:** Define a custom table (x_1297102_ip_geolo_ip_geolocation_data) to store comprehensive geolocation data.
- **API Integration:** Make an outbound REST API call to ipgeolocation.io to retrieve data.
- **Scripting:** Create a Server Script to handle the API call, parse the nested JSON response, and map the data to the correct table fields.
- **UI:** Design and implement a Service Portal widget with an input field, a button, and a dynamic table to display the results.
- **Results:** Test the outcome by verifying that the UI populates with data and a new record is created in the table after each lookup.
- **Conclusion:** Evaluate the success of the application's functionality and its readiness for deployment.

5. Project Design Phase

1. Create Table

- Open service now.
- Click on **All** >>System Definition >> search for **Tables**
- Click on **New**

servicenow All Favorites History Workspaces Admin **Table - IP Geolocation Data** Search

Table IP Geolocation Data Delete Update Delete All Records

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. [More Info](#)

* Label Application ?

* Name Remote Table ☐

Columns Controls Application Access

Table Columns for text Search 1 to 20 of 29 New

Column label	Type	Reference	Max length	Default value	Display
Updated	Date/Time	(empty)	40		false
Sys ID	Sys ID (GUID)	(empty)	32		false
Created by	String	(empty)	40		false
Created	Date/Time	(empty)	40		false
Updated by	String	(empty)	40		false
Updates	Integer	(empty)	40		false
Currency - Code	String	(empty)	40		false
Location - Latitude	Decimal	(empty)	20		false
Latitude	Decimal	(empty)	20		false
Location - Longitude	Decimal	(empty)	20		false
Location - Country Name	String	(empty)	40		false
State/Prov	String	(empty)	80		false
Location - Continent Name	String	(empty)	40		false

- Fill the following details to create a new Table

servicenow All Favorites History Workspaces Admin **Table - IP Geolocation Data** Search

Table IP Geolocation Data Delete Update Delete All Records

Sys ID	Sys ID (GUID)	(empty)	32		false
Created by	String	(empty)	40		false
Created	Date/Time	(empty)	40		false
Updated by	String	(empty)	40		false
Updates	Integer	(empty)	40		false
Currency - Code	String	(empty)	40		false
Location - Latitude	Decimal	(empty)	20		false
Latitude	Decimal	(empty)	20		false
Location - Longitude	Decimal	(empty)	20		false
Location - Country Name	String	(empty)	40		false
State/Prov	String	(empty)	80		false
Location - Continent Name	String	(empty)	40		false
Currency - Symbol	String	(empty)	40		false
Location - Country Code 2	String	(empty)	40		false
Currency - Name	String	(empty)	40		false
Location - State/Prov	String	(empty)	40		false
Timezone Name	String	(empty)	40		false
Country Name	String	(empty)	80		false
Longitude	Decimal	(empty)	20		false
Insert a new row...					

Delete Update Delete All Records

Related Links
[Form Builder](#)
[Design Form](#)

- Click on **submit**

2. Widget

- Click on **All >> System UI >> Widgets**
- Click on **New**, the form will open

The screenshot shows the ServiceNow interface for configuring a widget. The top navigation bar includes 'servicenow', 'All', 'Favorites', 'History', 'Workspaces', and 'Admin'. The current page is 'Widget - IP Geolocation Widget'. The configuration form includes the following fields:

- Name:** IP Geolocation Widget
- ID:** ip_geolocation_widget
- Description:** (empty)
- Application:** IP Geolocation Finder
- Public:** ☐
- Roles:** [Edit](#)
- Body HTML template:** A text area containing CSS code for styling the widget.
- CSS:** (empty)

The CSS code in the Body HTML template section is as follows:

```
<style>
@import url('https://fonts.googleapis.com/css2?family=Times+New+Roman:wght@400;500;600;700&display=swap');

:root {
  --card-background: rgba(255, 255, 255, 0.95);
  --text-primary: #1a2b2c;
  --text-secondary: #4a5568;
  --border-color: #e2e8f0;
  --button-primary: linear-gradient(135deg, #667eea, #764ba2);
  --button-hover: linear-gradient(135deg, #5a67d8, #6b46c1);
  --shadow-light: 0 8px 28px rgba(0, 0, 0, 0.12);
  --shadow-hover: 0 12px 28px rgba(0, 0, 0, 0.18);
}

/* Background with image */
body {
  margin: 0;
  padding: 0;
  font-family: 'Times New Roman', serif;
}

.container-app {
  font-family: 'Times New Roman', serif;
  display: flex;
  justify-content: center;
  align-items: flex-start;
}
```

- Fill the details
- Click Save or Submit
- Add your HTML code in the Body HTML Template section
- Add the Server-Side Scripting to call the ip geolocation api
- Enter the Client Script to call the server script when Look Ip Address button is clicked
- Click on Submit

3. Pages

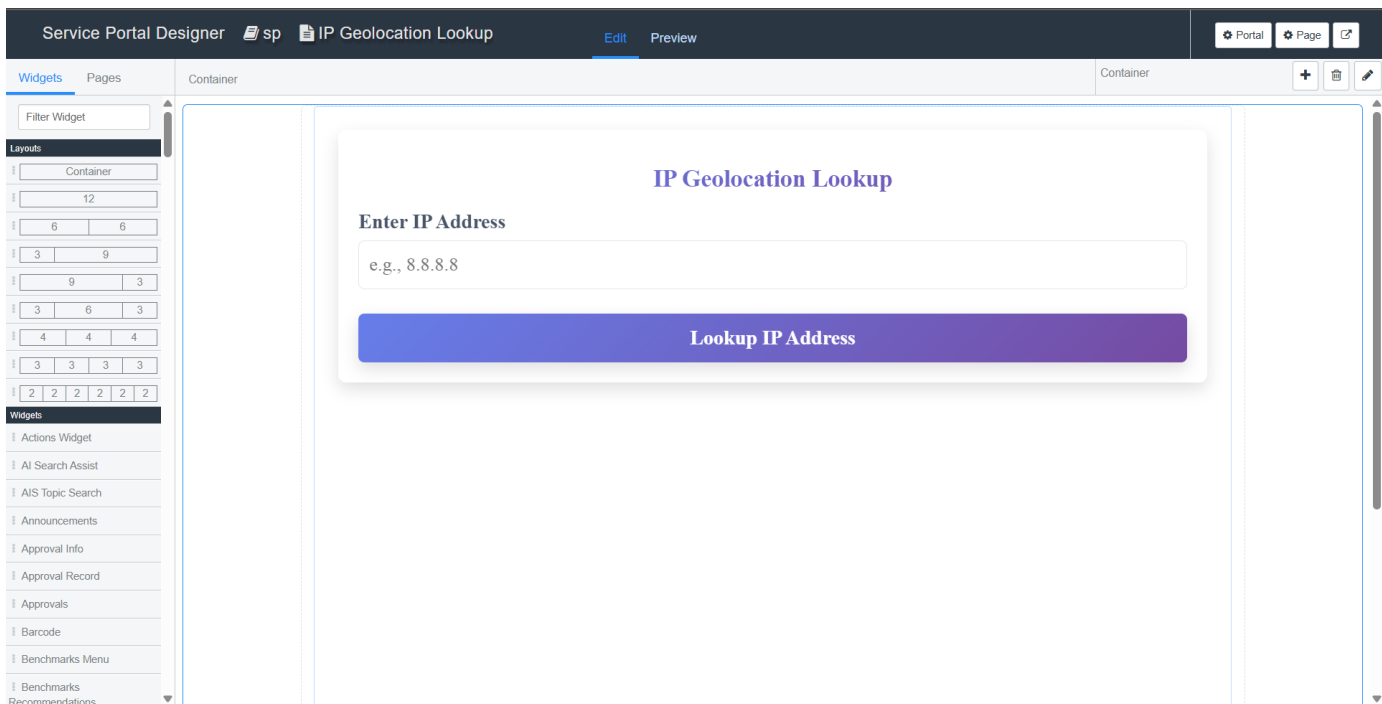
- Click on **All >>Service Portal >>Pages**
- Click on **New**, the form will open
- Enter the following details

The screenshot shows the ServiceNow interface for configuring a page. The top navigation bar includes 'servicenow', 'All', 'Favorites', 'History', 'Workspaces', and 'Admin'. The current page is 'Page - ip_geolocation_lookup'. The configuration form includes the following fields:

- Title:** IP Geolocation Lookup
- ID:** ip_geolocation_lookup
- Application:** IP Geolocation Finder
- Public:** ☐
- Draft:** ☐
- Roles:** [Edit](#)

At the bottom of the form, there are buttons for 'Try It', 'Update', 'Clone Page', and 'Delete'.

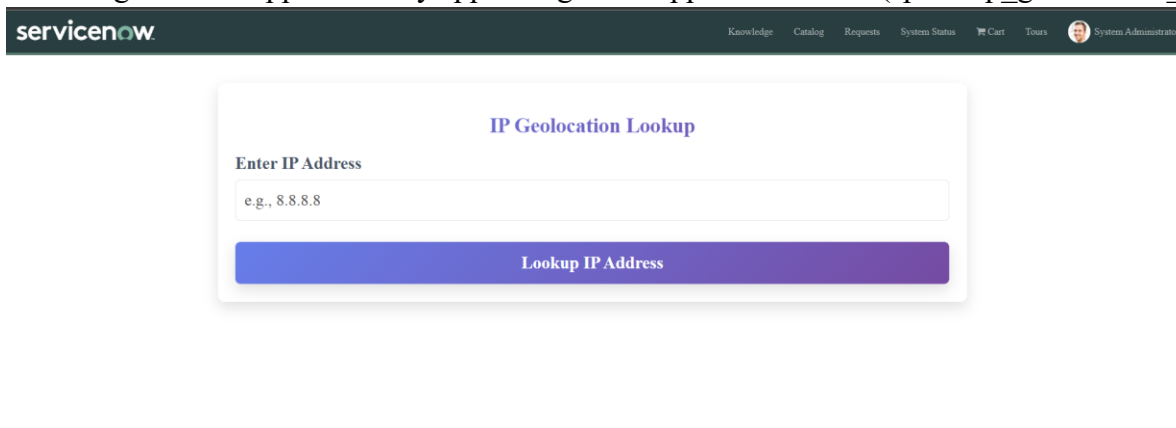
- Click Save or Submit
- Few Links will be activated at bottom
- Click on Open in Designer Link



- Choose a container from layout
- Select the Grid
- Add the Widget to the container by dragging and dropping

6. Performance & Functional Testing Phase

- Navigate to the application by appending this snippet to the URL (`sp?id=ip_geolocation_lookup`)



- Enter any valid Ip address and Click on Lookup IP Address button

servicenow

KnowledgeCatalogRequestsSystem StatusCartToursSystem Administrator

IP Geolocation Lookup

Enter IP Address

40.215.245.51

Lookup IP Address

Country Name

India

Country Code (2)

IN

State / Province

Telangana

City

Hyderabad

Zipcode

500029

Latitude

17.40650

Longitude

78.47724

Timezone

Currency

Indian Rupee (₹)


Calling Code

+91

IP Address

49.205.245.52

Country Flag



Raw JSON Response

- All the Geolocation Information is retrieved and displayed in the UI

7. Final Conclusion

The Geolocation Data Management Application demonstrates ServiceNow development skills in UI design, API integration, backend logic, and database management. It provides a practical, efficient, and user-friendly solution for IP-based geolocation lookups.