

Metro Ticket Generating System

Testing and Validation

Introduction

The testing phase validates that the **Metro Ticket Generating System**, including the Service Catalog, Flow Designer automation, and digital QR fulfilment, functions correctly according to business requirements. Testing was performed in a controlled ServiceNow environment using valid test data to ensure all core functional paths specifically station selection, fare calculation, and QR rendering were verified end-to-end.

Test Environment

The validation was conducted using the following environment configurations:

- **Platform:** ServiceNow.
- **Modules Tested:** Service Catalog and Flow Designer.
- **User Roles Involved:**
 - **Commuter (Requester):** Initiates the metro ticket booking or smart card recharge.
 - **IT Administrator:** Manages the backend data records and automation logic.

Test Execution Details

Testing focused on verifying that user inputs were accurately captured, fares were calculated dynamically, and records were stored securely without manual intervention.

Field Validation and Automation

- **Mandatory Fields:** Enforced on both the client and server side to prevent incomplete submissions, such as booking without a source or destination.
- **Auto-Populate Logic:** Catalog client scripts and UI policies correctly set values for single or return journeys and calculated the final fare in real-time.
- **Field Mapping:** Used the Process Automation Engine to ensure that values entered by commuters in the catalog variables were perfectly mapped to the corresponding fields in the custom database.

User Interface (UI) Validation Screenshots

The following images validate the dynamic behavior of the Service Catalog form based on user selection.

This screenshot shows the 'Book a Metro Ticket' form in its initial state. The primary selection variable, 'What do you want to do Today?', has the 'None' option selected. The 'Recharge Metro Card' and 'Book QR Ticket' options are available but not selected. On the right side of the form, there are input fields for 'Quantity' (set to 1), 'Delivery Time' (set to 2 Days), and buttons for 'Add to Cart' and 'Save as Draft'. Below the main form area is an 'Add attachments' section with a placeholder for file uploads. The top navigation bar includes links for Knowledge, Catalog, Requests, System Status, Cart, Tours, and System Administrator.

Figure 1: Default Form State (None Selected)

This view shows the initial state of the form where only the primary selection variable is visible to reduce initial clutter.

This screenshot shows the same 'Book a Metro Ticket' form after the 'Recharge Metro Card' option has been selected. The 'Recharge Metro Card' radio button is now checked. Below it, two new input fields have appeared: 'Enter Smart Card Number' containing 'CH000010901' and 'Enter Smart Card Name' containing 'METRO CITY 091'. The rest of the form, including the delivery options and attachment section, remains the same as in Figure 1. The top navigation bar is identical to the one in Figure 1.

Figure 2: Smart Card Recharge Selection

When "Recharge Metro Card" is selected, the form dynamically reveals card-specific variables like Smart Card Number and Name.

Figure 3: QR Ticket Booking Selection

When **Book QR Ticket** is selected, the form reveals journey station references and the auto-calculated fare preview.

Workflow and QR Execution

- 1. Workflow Trigger:** The Flow Designer logic was successfully triggered automatically immediately following the submission of a new ticket request.

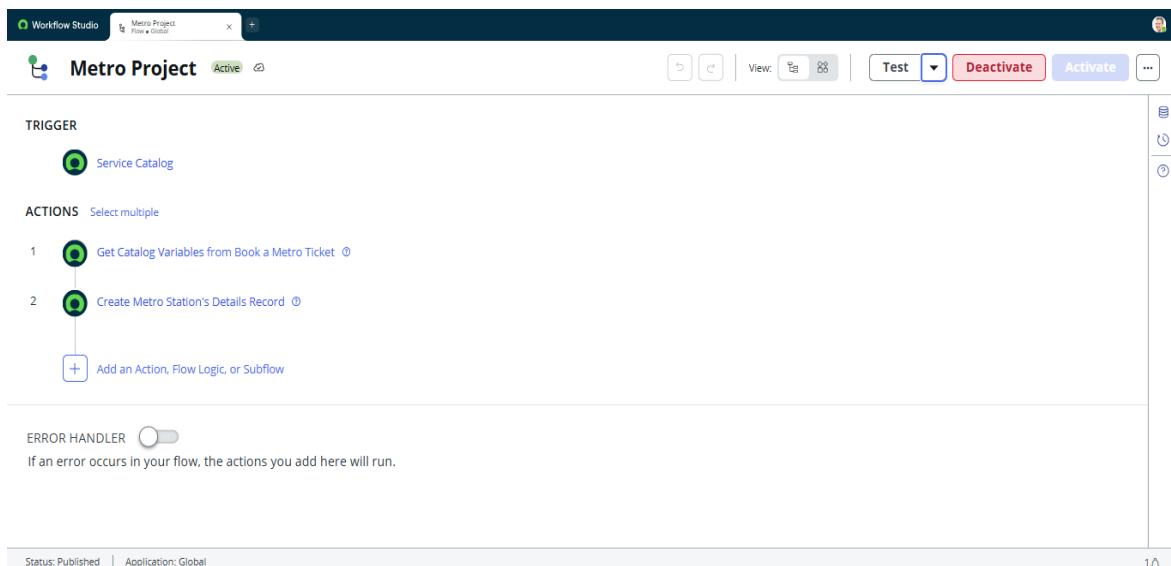
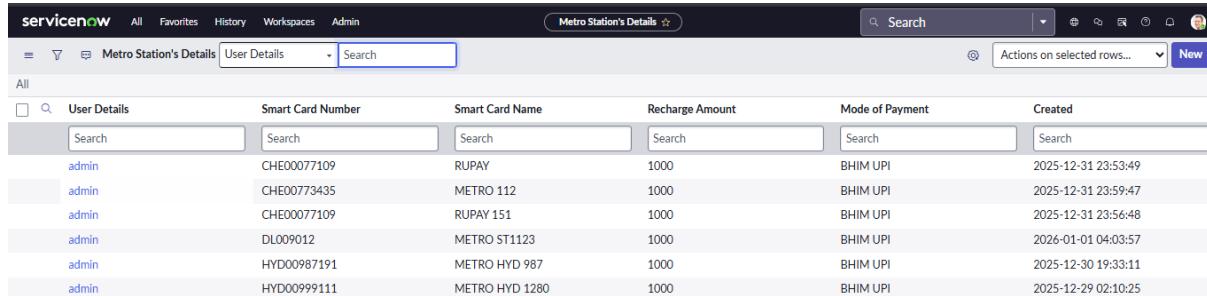


Figure 4: Flow Designer triggered automatically after Metro Ticket submission.

2. Record Creation: The system successfully created a structured record in the **Metro Database** (u_metro_station_s_details), ensuring data integrity.



The screenshot shows a ServiceNow interface titled "Metro Station's Details". The top navigation bar includes "servicenow", "All", "Favorites", "History", "Workspaces", and "Admin". Below the title is a search bar with "Search" and a dropdown menu "Actions on selected rows...". A "New" button is also present. The main area displays a table with columns: "User Details", "Smart Card Number", "Smart Card Name", "Recharge Amount", "Mode of Payment", and "Created". The table contains six rows of data, each with "admin" in the User Details column and various card numbers and names in the other columns. The "Created" column shows dates ranging from 2025-12-31 to 2026-01-01.

User Details	Smart Card Number	Smart Card Name	Recharge Amount	Mode of Payment	Created
admin	CHE00077109	RUPAY	1000	BHIM UPI	2025-12-31 23:53:49
admin	CHE00773435	METRO 112	1000	BHIM UPI	2025-12-31 23:59:47
admin	CHE00077109	RUPAY 151	1000	BHIM UPI	2025-12-31 23:56:48
admin	DL009012	METRO ST1123	1000	BHIM UPI	2026-01-01 04:03:57
admin	HYD00987191	METRO HYD 987	1000	BHIM UPI	2025-12-30 19:33:11
admin	HYD00999111	METRO HYD 1280	1000	BHIM UPI	2025-12-29 02:10:25

Figure 5: Database record generated with mapped travel variables.

3. QR Rendering: Upon submission, the onSubmit Catalog Client Script executed successfully, fetching the sys_id and rendering a unique scannable QR code via a modal popup for the passenger.

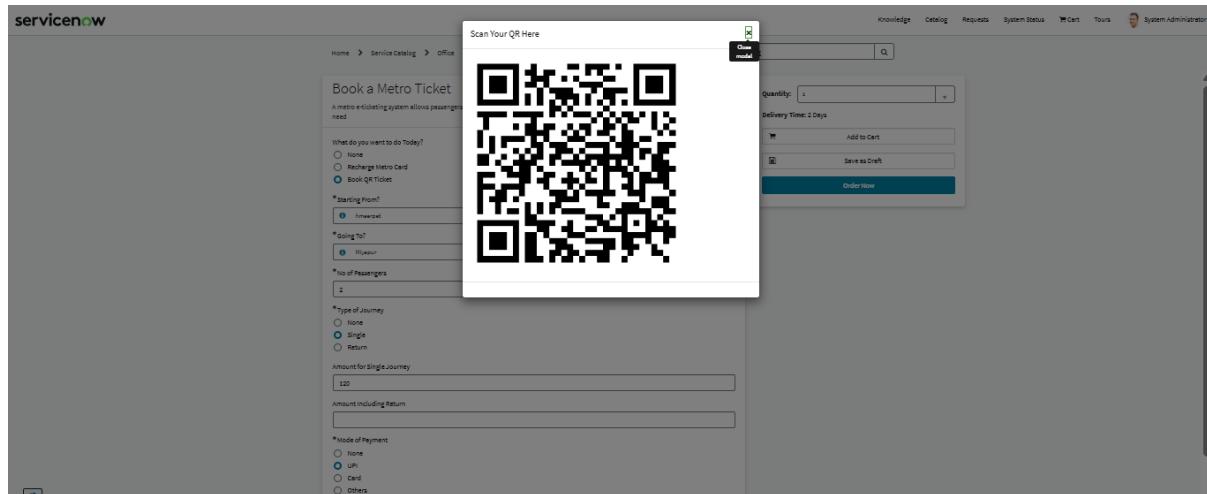


Figure 6: QR code ticket rendered for immediate traveler use.

4. Flow Completion: The execution details confirm that the backend lifecycle completed all actions from variable capture to record storage.

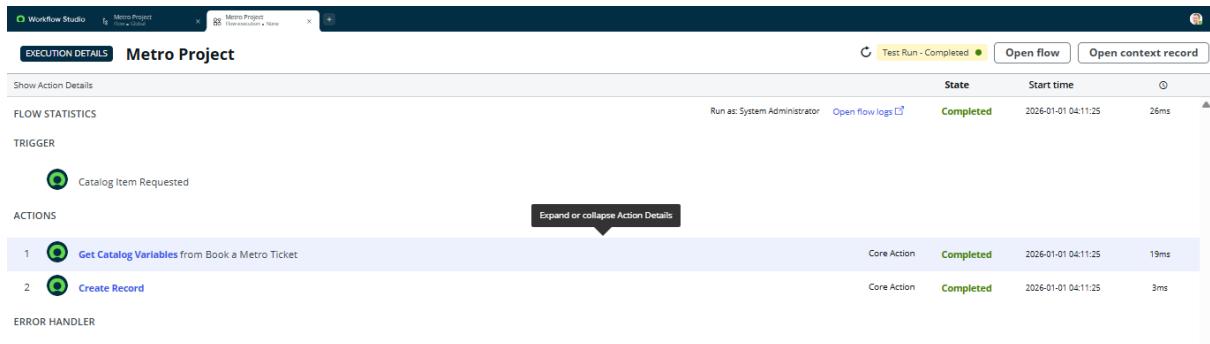


Figure 7: Flow execution completed successfully for Metro Project.

Security and Access Control (ACLs)

- **Role-Based Access:** Custom ACLs were verified to ensure that while commuters can book tickets or recharge cards, they cannot directly access or browse the backend database table.
- **Default Protections:** The four default ACLs created during table setup were utilized to maintain secure, role-restricted access data.

Access Controls (4) Security Data Filters Labels (1) Database Indexes (1) Table Subscription Configuration (1)						
<input type="checkbox"/> Updated <input type="button" value="Search"/> Actions on selected rows...						
Access Controls						
#	Name	Decision Type	Operation	Type	Active	Updated by
1	u_metro_station_s_details	Allow If	delete	record	true	admin
2	u_metro_station_s_details	Allow If	read	record	true	admin
3	u_metro_station_s_details	Allow If	create	record	true	admin
4	u_metro_station_s_details	Allow If	write	record	true	admin

Figure 8: ACL Verification

Conclusion

The testing results confirm that the **Metro Ticket Generating System** meets all functional requirements. The solution performs reliably across all critical stages request submission, dynamic fare calculation, and instant QR rendering ensuring a stable and efficient deployment for urban transit.