Project Report: Importing and Securing Data in ServiceNow

- Project Objective

The objective of this project is to import external data into the ServiceNow platform efficiently and securely. The project focuses on maintaining data integrity, preventing duplication, and implementing strong access control mechanisms through role-based access, table-level and field-level security, and audit logging.

- Data Import Process
-

2.1 Data Sources

Data was imported from multiple external formats and sources, including: CSV Files: Used for importing user and

asset records.
REST API Endpoints: Used for live data synchronization with third-party systems.
Excel Files: Used for one-time and bulk data updates.

2.2 Tools and Modules Used
Import Sets:
Module: System Import Sets → Load Data
Purpose: Uploads source files to temporary staging tables.
Transform Maps:
Module: System Import Sets → Create Transform Map
Purpose: Maps data fields from staging tables to target tables like sys_user and cmdb_ci.
Scheduled Imports:

Setup of scheduled jobs for recurring data import (daily/weekly).

2.3 Data Cleaning and Validation
Used transform scripts to standardize data (e.g., date formatting, trimming whitespace).
Conducted pre-import and post-import validations to ensure accuracy.
Used coalesce keys to avoid duplicate record creation.

- Data Security Implementation

3.1 Role-Based Access Control (RBAC)
Created custom roles (e.g., data_importer, cmdb_editor).
Mapped access permissions based on job responsibilities.
Used ACL rules from System Security →

Access Control to verify access.

3.2 Table and Field-Level Security
Restricted access to critical tables such as:
cmdb_ci (Configuration Items)
sys_user (User Records)
Sensitive fields (e.g., password, email, SSN) were protected using:
Field-level ACLs
Read/Write restrictions

3.3 Policies and Controls
UI Policies: Controlled field visibility and editability during form input.
Data Policies: Enforced server-side validation for data coming from imports or APIs.

3.4 Auditing and Logging
Enabled auditing for tables containing sensitive data.
Reviewed System Logs → Security for unauthorized access attempts.
Created dashboard widgets to track:
Import job success/failure
Security-related events

- Outcomes and Benefits

Successfully imported over [X] thousand records with 95%+ accuracy.
Reduced manual effort by 80% through automation.
Ensured secure data handling and compliance with governance rules.
Enabled continuous monitoring and security visibility through dashboards.

- Recommendations and Future Enhancements

Integrate MID Server for secure and real-time database imports.
Apply Data Classification to identify and tag sensitive records.
Enable Data Loss Prevention (DLP) and Encryption at Rest for critical information.
Periodically review roles and permissions using Access Control Review module.

- Appendices

Appendix A: Sample CSV Format
Appendix B: Screenshot of Transform Map
Appendix C: Sample ACL Script
// Allow users with 'itil' role to read incidents

```
gs.hasRole('itil');
```