Resume Builder Application – Complete Documentation

1. Application Overview

Scope: x_1554358_resume

Application Name: Resume Builder

Purpose:

A dynamic resume and cover letter management system that allows users to:

- Maintain a centralized professional profile
- Generate multiple tailored resume versions (ServiceNow, Desktop Support, General)
- Dynamically filter resume content based on job type
- Preview resumes and cover letters in formatted HTML
- Export structured resume data as JSON

---

2. Key Features

? Resume Type Filtering

- Resume Type controls what data appears in:
    - Skills
    - Projects
    - Certifications
- Enables one profile → multiple tailored resumes

? HTML Resume Preview

- Generates styled resume output
- Print/PDF friendly
- Built using ResumePreviewUtil

? JSON Export

- Structured data output
- Matches preview filtering logic
- Built using ResumeExportUtil

? Cover Letter Support

- Create and manage cover letters
- HTML preview generation
- Built using CoverLetterPreviewUtil

? Profile-Centric Design

- All data tied to a **Resume Profile**
- Auto-populated from sys_user
- Single source of truth

---

3. System Architecture

Core Relationship Model

Resume Profile (Parent)

↓

Resumes (Child – Resume Type lives here)

↓

Filtered Output (Preview / JSON)

Profile also connects to:

- Work Experience

- Skills

- Projects

- Certifications

- Education

- Cover Letters

4. Data Model

Resume Profile

- Full Name
- Email
- Phone
- Location
- Summary

Resume

- Profile (reference)
- Resume Type (ServiceNow, Desktop Support, General)
- Professional Title
- Summary Override
- Status

Skills

- Skill Name
- Include in:
    - ServiceNow Resume
    - Desktop Support Resume
    - General Resume

Projects

- Project Name
- Role
- Description

- Technologies
- URL
- Include flags (same as Skills)

Certifications

- Certification Name
- Issuing Organization
- Dates
- Include flags (same pattern)

Work Experience

- Job Title
- Employer
- Dates
- Description

(Not filtered — always included)

Cover Letter

- Profile
- Resume (optional)
- Company
- Job Title
- Letter Body

---

5. How Filtering Works

When generating a resume:

1. The system reads:
    - Resume record

- Resume Type

2. For each section (Skills, Projects, Certifications):
    - A GlideRecord query is executed
    - A filter is added based on Resume Type

Example:

```
if (resumeType == 'ServiceNow') {
   skillGR.addQuery('include_in_servicenow_resume', true);
}
```

3. Only matching records are included in:
    - HTML preview
    - JSON export

---

6. Script Includes

ResumePreviewUtil

- Generates HTML resume output
- Applies Resume Type filtering
- Handles formatting and layout

ResumeExportUtil

- Builds JSON object
- Mirrors preview filtering logic
- Used for data export

CoverLetterPreviewUtil

- Generates HTML cover letter preview
- Includes profile + company/job context

---

7. User Workflow

Step 1: Create Profile

- Auto-populated from user
- Add summary, contact details

Step 2: Add Data

- Work Experience
- Skills
- Projects
- Certifications

Mark items for appropriate Resume Types.

Step 3: Create Resume

- Select Resume Type:
    - ServiceNow
    - Desktop Support
    - General

Step 4: Preview Resume

- Click **Preview Resume**
- Generates HTML attachment
- Can print → PDF

Step 5: Export JSON

- Click **Export JSON**
- Outputs structured data matching preview

Step 6: Create Cover Letter

- Add company/job details
- Reuse base template
- Preview via button

---

8. Design Decisions

Why Resume Type lives on Resume

- Allows multiple versions per profile
- Keeps profile clean and reusable

Why Work Experience is not filtered

- Resume history should remain complete
- Avoids unrealistic resume gaps

Why filtering uses checkboxes

- Simple
- Flexible
- Easy to maintain

---

9. Future Enhancements (Phase 2)

- Service Portal landing page
- Record Producers for Resume/Cover Letter
- “My Resumes” and “My Profile” portal views
- Resume duplication feature
- XML export
- UI polish and layout improvements

Technical Reference (Option B)

Core Logic Pattern

Filtering Pattern (used everywhere)

```
if (resumeType == 'ServiceNow') {
   gr.addQuery('include_in_servicenow_resume', true);
}
```

Preview Generation Flow

1. UI Action triggered
2. Calls Script Include
3. GlideRecord retrieves:
    - Resume
    - Profile
4. Related tables queried with filters
5. HTML string constructed
6. Stored as attachment

JSON Export Flow

1. UI Action triggered
2. Script Include builds object
3. Same filtering logic applied
4. Object serialized to JSON
5. Returned/downloaded

Key Functions

esc()

function esc(value)

- Prevents HTML injection
- Escapes special characters

---

sectionTitle()

function sectionTitle(title)

- Standardizes section headers
- Keeps formatting consistent

Important Fields

Table	Field	Purpose
Resume	resume_type	Controls filtering
Skills	include_in_*	Filtering flags
Projects	include_in_*	Filtering flags
Certifications	include_in_*	Filtering flags

Debug Tips

- If preview fails → check Script Include syntax
- If filtering fails → verify field names
- If data missing → check checkbox flags
- If redirect happens → usually script error

Performance Notes

- All filtering done at query level (efficient)

- Minimal client-side processing
- Scales well for personal use

Resume Builder Application – Complete Documentation

---

1. Application Overview

Application Name: Resume Builder
Scope: x_1554358_resume
Platform: ServiceNow (Scoped Application + Service Portal)

Purpose

The Resume Builder application is a custom ServiceNow solution designed to:

- Manage a centralized professional profile
- Generate multiple tailored resume versions
- Create and manage cover letters
- Provide a user-friendly Service Portal interface
- Dynamically filter resume content based on job type
- Export structured resume data

---

2. Key Features

? Dynamic Resume Generation

- One profile supports multiple resume versions
- Resume Type determines content visibility

? Resume Type Filtering

Supports:

- ServiceNow
- Desktop Support
- General

Controls:

- Skills
- Projects
- Certifications

? HTML Resume Preview

- Styled, print-friendly layout
- Exportable as PDF
- Generated via Script Include

? JSON Export

- Structured resume data
- Matches preview filtering logic

? Cover Letter Management

- Create reusable cover letters
- Customize per company/job
- Preview support

? Service Portal Interface

- Clean landing page
- User-focused navigation
- Reduced reliance on backend forms

---

3. System Architecture

Data Model Structure

Resume Profile (Parent)

↓

Resumes (Child - defines Resume Type)

↓

Dynamic Output (Preview / JSON)

Related Tables

Profile connects to:

- Work Experience
- Skills
- Projects
- Certifications
- Education
- Cover Letters

---

4. Portal Architecture

Landing Page (resume_home)

Central navigation hub with:

- My Profile
- My Resumes
- Create Resume
- My Cover Letters
- Create Cover Letter

Pages Created

Page	Purpose
resume_home	Main landing page
my_profile	Access user profile
my_resumes	List of user resumes
resume_detail	Single resume view
my_cover_letters	List of cover letters
cover_letter_detail	Single cover letter view

Navigation

- Added to portal header via sp_menu_item
- Direct access from any page

---

5. Record Producers

Create Resume

- Table: Resume
- Fields:
    - Resume Type
    - Professional Title
    - Summary Override

Script:

- Automatically links Resume to user’s Profile

Create Cover Letter

- Table: Cover Letter
- Fields:
    - Company
    - Job Title
    - Letter Body

Script:

- Automatically links to Profile

---

6. Resume Type Filtering Logic

Core Concept

Resume Type acts as a filter layer applied during:

- Preview generation
- JSON export

Example Logic

```
if (resumeType == 'ServiceNow') {
   gr.addQuery('include_in_servicenow_resume', true);
}
```

Tables Using Filtering

Table	Fields
Skills	include_in_*
Projects	include_in_*
Certifications	include_in_*

Design Decision

Work Experience is **not filtered** to preserve full employment history.

---

7. Script Includes

ResumePreviewUtil

- Generates HTML resume
- Applies filtering logic
- Handles layout and formatting

ResumeExportUtil

- Builds JSON object
- Mirrors preview filtering
- Ensures consistency

CoverLetterPreviewUtil

- Generates HTML cover letter
- Uses profile + job/company context

---

8. Resume Detail Page

Displays:

- Resume number
- Resume type
- Professional title
- Status
- Profile
- Summary override

Provides actions:

- Edit Resume
- Preview (via backend)
- Export JSON

---

9. Cover Letter Detail Page

Displays:

- Company
- Job title
- Letter body
- Profile
- Linked resume (optional)

Provides actions:

- Edit Cover Letter
- Preview

---

10. User Workflow

Step 1: Create Profile

- Auto-populated from user
- Add personal and professional info

Step 2: Add Supporting Data

- Skills
- Projects
- Certifications
- Work Experience

Assign items to appropriate Resume Types.

Step 3: Create Resume

- Select Resume Type
- Add optional overrides

Step 4: Preview Resume

- Generates formatted HTML
- Can be printed as PDF

Step 5: Export JSON

- Outputs structured data
- Matches preview

Step 6: Create Cover Letter

- Enter company and role
- Customize message

---

11. Design Decisions

Profile-Centric Model

- Avoids duplication
- Enables reuse across resumes

Resume Type on Resume Table

- Allows multiple versions per user
- Keeps profile clean

Checkbox-Based Filtering

- Simple implementation
- Highly flexible

Portal-First UX

- Reduces backend dependency
- Improves usability

---

12. Technical Notes

Key Tables

Table	Purpose
Resume Profile	User data
Resume	Resume definitions
Skills	Skill records
Projects	Project records
Certifications	Certification records
Cover Letter	Cover letters

Debug Tips

- Redirect issues → Script Include errors
- Missing data → Check include flags
- Profile issues → Verify user linkage
- Portal issues → Check page ID and sys_id parameters

Performance

- Query-level filtering
- Minimal client-side processing
- Efficient for intended use

---

13. Future Enhancements

Phase 3 Ideas

- Portal-based Preview (no backend dependency)
- Duplicate Resume feature
- Resume version comparison
- Rich text editor improvements
- Export to PDF directly
- Role-based sharing

---

14. Portfolio Value

This application demonstrates:

- Data modeling
- Script Includes
- GlideRecord usage
- Service Portal development
- UX design principles
- Real-world problem solving

---

15. Summary

The Resume Builder application transforms ServiceNow into a:

- Personal productivity tool
- Resume generation engine
- Portfolio-ready application

It showcases both **technical capability** and **practical design thinking**.