
Product Requirements Document (PRD)

Product: Job Description Builder (MVP)

Target Users: Lean employers (SMEs, startups, nonprofits)

Primary Goal: Enable employers to create high-quality job descriptions in under 10 minutes without prior HR expertise.

---

1. Product Overview

Problem

Lean employers often lack HR expertise and time to create structured, effective job descriptions. This leads to:

- Poor quality job postings
- Longer hiring cycles
- Poor candidate matching
- Hiring friction

MVP Objective

Provide a structured, template-driven job description builder that:

- Eliminates blank page creation
- Guides employers step-by-step
- Enables fast, professional job description creation
- Supports draft saving and publishing

---

2. Core MVP Features

Feature 1: Template Library

Functional Description

System shall provide a library of pre-defined job description templates.

Templates shall be selectable by:

- Industry
- Job Tile
- Seniority Level

Selecting a template creates a new editable Job Description Draft populated with template content.

Functional Requirements

FR-1.1 Template Storage

System shall store templates in database.

Template table schema:

templates
---
template_id (UUID, PK)
template_name (string, required)
industry (string, required)
job_title (string, required)
seniority_level (string, required)
employment_type (string, required)
job province
job city

job_descrption (text)
responsibilities (JSON array of strings)
qualifications (text)
Lowest Compensation
Highest Compensation
created_at (timestamp)
updated_at (timestamp)
status (active/inactive)

NOTE; other than Lowest Compensation
Highest Compensation, all the other fields should already exist in the database. Please add the two compensation fields to other job related screens

---

FR-1.2 Template Retrieval

System shall allow filtering templates by:

- Industry
- Job Tile
- Seniority Level

System shall return a list of matching templates.

FR-1.3 Template Selection

When user selects a template:

System shall create new Job Description Draft using template data.

New record shall be created in the job_descriptions table.

User Benefit

- Eliminates blank page problem
- Enables instant structured job description creation

Feature 2: Multi-Step Job Description Builder Wizard

Functional Description

System shall guide user through structured job description creation via multi-step wizard.

Wizard Steps

Step 1: Job Basics

Fields:

- Job Title (required)
- Industry (required)
- Job Name (required)
- Seniority Level (required)
- Employment Type (required)
- Work Location Type (Remote / Hybrid / Onsite)
- Location City (optional)

Step 2: Job Description

Fields:

- Job Description (required, text)

Step 3: Responsibilities

Fields:

- Responsibilities list (required)
- User can:

    - Add responsibility
    - Edit responsibility
    - Delete responsibility

Minimum: 1 responsibility required

Step 4: Skills Selection

Fields:

- Required Skills (required, multi-select)
- Good to have Skills (optional, multi-select)

NOTE: let's update database and excel to have separate fields for required vs good to have skills

---

Step 5: Qualifications and Compensation

Fields: all mandatory

- ☐ Qualifications
- ☐ Compensation Range Minimum
- ☐ Compensation Range Maximum
- ☐ Compensation Currency (display only - default CAD)

Step 6: Review and Save

User can:

- Save Draft
- Publish Job Description

Feature 3: Skills Database and Multi-Select

Functional Description

System shall provide a searchable static skills database.

Skills Table Schema

skills
------
skill_id (UUID, PK)
skill_name (string, required)
skill_category (string, optional)
status (active/inactive)
created_at (timestamp)

Many-to-Many Relationship Table

job_description_skills
----
id (UUID, PK)
job_description_id (FK)
skill_id (FK)
skill_type ("required" or "preferred")
created_at (timestamp)

Functional Requirements

User shall be able to:

- Search skills via text input
- View matching results
- Select multiple skills
- Remove skills

Feature 4: Job Description Draft Management

Job Description Table Schema

job_descriptions
---
job_description_id (UUID, PK)
user_id (FK, required)
template_id (FK, optional)
job_title (string, required)
industry (string)
job_function (string)
seniority_level (string)
employment_type (string)
location_type (string)
location_city (string)
job_description (text)
qualifications (text)
compensation_min (decimal)
compensation_max (decimal)
compensation_currency (string)
status ("draft", "published")
created_at (timestamp)
updated_at (timestamp)
published_at (timestamp, nullable)

Feature 5: Draft Saving

Functional Requirements

User shall be able to:

- Save job description as Draft
- Exit builder

- Resume editing later

Draft shall persist all entered data.

Draft shall be associated with user_id.

Feature 6: Publish Job Description

Functional Requirements

Users shall be able to publish job descriptions.

When published:

- status = "published"
- published_at timestamp set

Published jobs shall be visible on job hub.

Draft jobs shall NOT be visible publicly.

Feature 7: Edit Existing Job Description

Functional Requirements

User shall be able to edit:

- Draft job descriptions
- Published job descriptions

When published job is edited and saved:

Option selected for MVP (recommended):

Save changes as Draft version.

Users must republish.

Feature 8: Duplicate Job Description

Functional Requirement

Users shall be able to duplicate an existing job description.

System shall:

- Create new job description record
- Copy all fields
- Set status to draft
- Assign to same user

Feature 9: Validation Rules

Required Fields Validation

Field	Requirement
Job Title	Required, pick from drop down, when “Others” is selected to have afield to enter the actual value, min 3 characters
Job Description	Required, min 50 characters
Responsibilities	Minimum 1 required
Required Skills	Minimum 1 required
Compensation Range Minimum	Required, numeric

Compensation Range Maximum	Required, numeric

Validation Behavior

System shall:

- Display inline validation messages
- Prevent publish if validation fails
- Allow saving draft even if incomplete

Feature 10: User Ownership and Access Control

Functional Requirements

Each job description shall be associated with user_id.

User shall only be able to:

- View own job descriptions
- Edit own job descriptions
- Delete own job descriptions

Feature 11: Delete Job Description

Functional Requirement

Users shall be able to delete job descriptions.

System shall perform soft delete.

Add field:

deleted_at (timestamp, nullable)

Deleted records not shown to user.

Feature 12: Employer Job Dashboard

Functional Requirements

Users shall see a list of their own job descriptions.

Display:

- Job Title
- Status (Draft / Published)
- Last Modified Date

User actions:

- Edit
- Delete
- Duplicate
- Publish / Unpublish

---

3. Non-Functional Requirements

Security

Users shall only access own records.

All API endpoints require authentication.

Scalability (MVP expectation)

Support:

- 10,000 employers
- 100,000 job descriptions
- 10,000 skills

---

4. Explicit MVP Scope Inclusion / Exclusion

Included in MVP

Template Library
Multi-Step Wizard
Draft Saving
Publish / Unpublish
Skills Multi-Select
Skills Database
Edit Job Description
Duplicate Job Description
Validation
Dashboard

Excluded from MVP

AI job description generation
AI skill suggestions
Autosave
Candidate matching
Analytics

Employer branding customization
Approval workflows

Performance:

- System shall load the builder in < 2 seconds.
- Template retrieval < 1 second.
- Skill search response < 500 ms.

---

5. Complete User Flow

Step 1: User logs in

Step 2: User clicks "Create Job Description"

Step 3: User selects template OR starts blank

Step 4: System creates draft

Step 5: User completes wizard

Step 6: User saves draft OR publishes

Step 7: Job appears on job hub if published

---

6. Workflow to Test: Acceptance Criteria

AC-1
Given user selects template
When template selected
Then draft job description created

AC-2
Given required fields missing

When user clicks publish
Then system prevents publish and shows validation errors

AC-3
Given user saves draft
When user returns
Then draft loads with saved data