Capstone Progress Report

Canon EMEA - Matching Employees to Projects

Saad Joiya

# Table of Contents

* Overview
* Status and Next Steps
* Required Data Tables and Features
* Mock Data Creation: Design and Strategy
* Matching & Scoring Criterions

Overview **Partner**Canon – Europe Middle East and Africa

Canon provides state-of-the-art imaging solutions to its clients which are spread over a vast array of domains. It has around 150,000 employees across the globe

**Problem Statement**Currently, HR of Canon EMEA uses a manual process to match suitable employees for incoming projects. This process is not optimal considering time, effort, scalability and human errors. Moreover, A streamlined framework for data collection is also currently absent – employee and project features are being hand filled.

**Project Motivation**  
For an organization of Canon’s size and stature it becomes imperative to optimally match employees and projects to maximize efficiency and enhance delivery. This project would reduce a lot of workload from HR and other managers who can easily find the best employees for different project needs. Moreover, we can enhance the quality of employee allocation by more accurate and consistent recommendations based on matching features. This system can be improved over time with innovations as this is proof of concept with less prior work done in this regard.

**Project aim**  
We are trying to create an algorithm to automatically output a list of ideal employees based on project requirements.

**Method**  
Employee and Project Feature Space:   
Define the features that would be used in both the employees and projects table. Next, we would create mock data with relevant data format that mimics real world data closely.

Score Calculation:  
We need to figure out which features do we need to match from both tables to generate relevant scores, what matching techniques do we need to consider in order to optimally generate scores for these subgroups and what weight should we give to different feature groups in final scoring? Next, we use some features in post processing for filtering optimal employees.

|  |  |
| --- | --- |
| ***What do I need?*** | ***Where do I get it from?*** |
| *Employees Data* | *Mock data creation based on client recommendation* |
| *Projects Data* | *Mock data creation based on client recommendation* |
| *Data Fields for both tables* | *Project Sponsor from Canon* |
| *Data Types for all features* | *Project Sponsor from Canon* |
| *Relevant values for features in both tables* | *Online research* |
| *Algorithm for matching relevant fields* | *Algorithms research* |
| *Feedback on match scores* | *Arena Approach (collaboration with project sponsor)* |

**Limitations**

* We do not have any historic data available. We have to create it ourselves based on the feature set recommended by the client.
* We do not have any true labels so we cannot train an ML model, so we need to use statistical approaches. Moreover, the validity of our algorithm’s output scores would be based on project sponsors’ judgement.
* Human errors can be present during data collection as well as scoring as some of the data fields are hand filled by humans. Moreover, perception bias can also skew results.

**Disadvantages**  
A lot of research and manual work is needed in data creation and algorithm implementation. This model cannot learn with new data as this is not an ML model.

**Advantages**  
High interpretability. We can use weights as needed to control how the employees are ranked, and which element contributes more than others. This method is not black box so we can tweak based on client expectations. We can also apply post filtering based on employee availability etc., that is a plus.

**Output Format**For easy reproducibility and collaboration we generate the following for this project:

Capstone Report: Contains project details, complete overview of data, matching and scoring mechanism and additional information

Jupyter Notebook: Complete process from Data Creation to Final Scoring would be implemented in a Jupyter notebook containing all dependencies and explanations

Backup and storage: Storing the report and the code in a GitHub repository, shared with the project sponsor of the company

**Suggestions**  
Canon Learn and Development team should try to gather project and employee profiles in the structure described for using this matching algorithm. As data starts to accumulate, we can test the efficacy of these matchings. With time, as we gather performance labels, we can use the features from projects and employees as predictors to train classification models which automate the matching process. Furthermore, using this model as the baseline we can further innovate and improve the processes as more data starts to get collected. This data can also be used in other company-wide analysis.

**Evaluation Criterion**  
The Arena approach is implemented for now; the results are periodically shared with the client. The feedback received on matches is then used to optimize the algorithm.

Status and Next Steps

**Status**

* Collaborated with the client to list all the features for employees and projects table.
* Researched the internet to collect insights into different types of themes/departments prevalent in Canon.
* Collected Canon specific information on different features based on these different themes for mocking real world data.
* Researched different matching algorithms that can be used based on feature values and their data types.
* Was able to generate projects and employee’s data for all features except features that would be used in post process filtering and bonus scoring.
* This data closely mimics real world data that would be collected and is internally consistent among instances.
* Figured out matching features in Projects and Employee tables and how they would interact to create scoring for subgroups.
* A first draft of scoring for all of the features has been implemented.
* Created a jupyter notebook with all required steps from data creation to scoring as per the current progress.
* Created this capstone report outlining all steps, processes and outcomes as per the current progress.

**Next Steps**

* Consult the client about improvements in the matching and scoring algorithm.
* Test the algorithm by generating higher volume of instances and their run times (Currently working for 20 projects and 20 employees = 400 scorings)
* Try other methods to improve matches in text-based fields.
* Implement code for interactively weighing different subgroups for final scoring.
* Iterate and improve the code and report with passing checkpoints.

# Required Data Tables and Features

We require two tables, namely Projects and Employees, to execute our algorithm. The features, data types and details for these tables were thoroughly discussed with Canon-EMEA.

**Project Table Details:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Description** | **Data Type** | **Details** |
| Project Summary | Give a summary of the project | Text | Free text, ex. "This project will support the European patent attorney team by building internal systems and support processes aligned with their strategic objectives." |
| Scope and Deliverables | Outline the scope and list milestones | Text | Free text, ex. "Design support processes, create documentation for best practices, collaborate with business units, and track implementation progress." |
| Customer Industry | Specify the customer’s industry (e.g., finance, healthcare) | List | Finance, Healthcare, Retail etc |
| Customer Preferences or Standards | Any specific standards, compliance, or customer communication preferences | Text | Free text or nonfinite list, ex "Needs to be compliant with ABC100 standard" or "XYZ project methodology is a must" |
| Products Involved | The specific solutions the project centers on, which is crucial if certain employees have experience in those | List | List. For e.g. MVP AI SCAN, WORKFLOW2000, PRINT2.0 |
| Integration Requirements | Any required integrations with existing systems, third-party tools, or APIs | Text | Integration with ERP System, Master data exchange with rest API etc. |
| Required Skills and Expertise | Skill Requirements for the project e.g., JavaScript, project management, Printing | Dictionary | JavaScript, Python, Project Management, Graphic Designing, and expertise level (1 to 10 expertise level) |
| Complexity Rating | A subjective rating classifies the project’s complexity | Integer | 1 to 10  ***Note****: This value is highly subjective based on who enters the data. Junior employees can underestimate this value whereas experienced employees can consider different complications to make it more realistic. During data collection we should use something like "Planning Poker" (reference to Scrum), with group estimates based on a reference project.* |
| Work Location | Location of Work Office | Category | Canon Offices Cities/European City |
| Work Flexibility | Indicate whether the project can be remote, hybrid, or if it requires on-site work in specific locations | Category | Value from either of onsite, Remote, Hybrid |
| Language Requirements | Languages required to communicate in this project | Dictionary (key) | Languages like English, German, French, Portuguese etc. (nonfinite list can put everything else in others) |
| Language Level | CEFR Level of languages required to communicate in this project | Dictionary (value) | A1, A2, B1, B2, C1, C2 |
| Effort | Estimated workload in hours required for the project | Integer | Estimated hours required to complete the project |
| Requested End | Desired End Date | Date | Desired date for project completion |

**Employee Table Details:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Description** | **Data Type** | **Details** |
| Role | Job Description | Text | For e.g. "Provides high-quality patent services including drafting, filing, and prosecuting patents" |
| Industry Experience | Specify industries experienced in | List | Finance, Healthcare, Retail etc |
| Internal /External Certifications | Relevant certifications (e.g., PMP, Six Sigma). | List | For e.g. PMP, Six Sigma certifications etc. |
| Product Experience | Canon Product Names, redacted | List | List. For e.g. MVP AI SCAN, WORKFLOW2000, PRINT2.0 |
| Expertise | Define expertise of employee | List | For e.g. i.e. Scripting, Integration, Color Reproduction, Cloud & Infrastructure |
| Core Competencies | Specific skills or technologies, e.g., programming languages, project management methodologies. | Dictionary (key) | For e.g. JavaScript, Python, Project Management, Graphic Designing |
| Core Competencies (Expertise) | Rating Specific skills or technologies, e.g., programming languages, project management methodologies. | Dictionary (value) | Core Competencies expertise level (1 to 10 expertise level) |
| Work Location | Location of Work Office | Category | Canon Offices Cities/European City |
| Work Flexibility | Indicate whether the project can be remote, hybrid, or if it requires on-site work in specific locations | Category | Value from either of Onsite, Remote, Hybrid |
| Languages Known | Languages required to communicate in this project | Dictionary (key) | Languages like English, German, French, Portuguese etc (nonfinite list can put everything else in others) |
| Communication Skills | CEFR Level of languages required to communicate in this project | Dictionary (value) | A1, A2, B1, B2, C1, C2 |
| Available From | Availability Start Date | Date | Date from which employee has working bandwidth |
| Weekly Availability in Hours | Hourly Bandwidth per Week | Integer | The number of hours employee is free to work per week |
| Cultural Awareness | Openness to diversity and experience in international settings, valuable for global teams. | Integer | Openness Rating 1 to 10 |
| Problem Solving | Critical thinking and adaptability in projects | Integer | Past Project Diversity Rating 1 to 10 |
| Leadership | Leadership or mentoring/coaching experience. | Integer | Past Experience Rating 1 to 10 |

# Mock Data Creation: Design and Strategy

To build a robust, realistic project-to-employee matching system, we developed a structured approach for generating mock data that mimics real-world conditions at a company like Canon EMEA. This data serves as the foundation for testing our feature-matching algorithms and scoring logic.

**1. Goal**

Generate two high-quality datasets:

* Projects Table: Represents incoming client projects Canon may handle
* Employees Table: Represents internal talent pool with skillsets and preferences

Each row in both datasets is rich in features relevant to HR and resource allocation.

**2. Role and Project Research**

To mimic Canon EMEA's context:

* We analyzed job postings on LinkedIn and Canon’s career portal
* From these, we curated 20 unique roles with concise, theme-tagged descriptions
* We also created 20 realistic project summaries, each with scope and business objectives

These were used in:

* Creation of Project Summary and Scope & Deliverable Descriptive Texts in Projects table.
* Creation of Role Name and Role Description in Employees table
* Guiding all relevant features for mocking the real world internally consistent data as discussed in point 3.

**3. Realism and Consistency**

To avoid random or inconsistent data, we imposed several real-world design constraints. We call this Thematic Control. This is done to ensure internal consistency between fields of the same project/employee. For example, if the project requires a person from HR, required skills would have values such as Talent Management and required certifications would have values such as PMP. Whereas, if a technical resource is required the skills might include Data Analysis, cloud services etc. and required certifications would have values such as Microsoft Azure Certification.

We bring internal consistency and realism using this approach.

We categorized all roles and projects under six high-level business themes:

* Technical
* Sales
* Marketing
* HR
* Legal
* Consulting

This ensured internal consistency between:

* Products involved
* Required skills and Expertise
* Customer Preferences (Certifications)
* Integration Requirements (Expertise Areas)
* Project Summary
* Scope and Deliverables

Each theme had predefined pools of realistic values.

***Note****: These feature names are benchmarked from projects table. We might have matching columns in the employee’s table but with different names. For information on theme specific values, refer to the* ***appendix.***

**4. Generic Pools**

Unlike theme specific pools that derive feature values based on themes for consistency and realism in data, some of the features are more generic. We designed the following generic vocabularies:

| **Category** | **Description** |
| --- | --- |
| Work Location | Main European cities like Berlin, Vienna, London etc. |
| Work Flexibility | Working options like remote, onsite or hybrid |
| Languages Required | European languages like English, French, Italian, German etc. |
| Language Level | Selected from CEFR levels (A1 to C2) |
| Industries | Canon-relevant verticals (Finance, Retail, etc.) |

**5. Human-Like Data Variability**

To simulate real-world messiness:

* Projects include intentional typos (e.g., "Brlin" for "Berlin") in fields like:
  + Work location
  + Products Involved
  + Customer Industry
  + Language Required
  + Required Skills and Expertise
* Employees remain clean (mirroring HR system records)

This forces the matching logic to rely on fuzzy similarity algorithms.

**6. Scalar Fields in Data**

| **Table** | **Feature** | **Format / Data Type** |
| --- | --- | --- |
| Projects | Expertise Value in Required Skills & Expertise (dictionary value) | Rating: 1 to 10 |
| Projects | Complexity | Rating: 1 to 10 |
| Projects | CEFR of Languages Known (dictionary value) | A1 to C2 |
| Projects | Effort | Int: Hours Required |
| Projects | Requested Timeline | Date: Delivery Date |
| Employees | Expertise of Core Competencies (dictionary value) | Rating: 1 to 10 |
| Employees | CEFR of Languages Known (dictionary value) | A1 to C2 |
| Employees | Cultural Awareness | Rating: 1 to 10 |
| Employees | Problem Solving | Rating: 1 to 10 |
| Employees | Leadership | Rating: 1 to 10 |
| Employees | Collaboration | Rating: 1 to 10 |

**7. Volume and Scalability**

* We generated 20 projects and 20 employees.
* Each entry is theme-aware, realistically structured, and fuzzily variable.
* The system is extensible: it can generate 1000+ rows with identical consistency.

**Outcome**

This mock data strategy gives us:

* High realism incorporating all relevant features with human errors in data collection
* Controlled variability
* Full feature coverage for testing matching models
* Easy extensibility for demos or ML-based learning systems

# Matching and Scoring Criterions

The following table gives the complete summary of the matching criterions between projects and employees data sets and the scoring methods. We describe each feature matching in detail in the next section.

|  |  |  |  |
| --- | --- | --- | --- |
| **Project Feature** | **Employee Feature** | **Matching Method** | **Notes** |
| Project Summary | Role | Text Embedding / Cosine Similarity | General fit & thematic similarity based on text embeddings and cosine similarity |
| Scope and Deliverables | Role | Text Embedding / Cosine Similarity | Task alignment based on text embeddings and cosine similarity |
| Customer Industry | Industry Experience | Category Similarity (Fuzzy match with threshold) | Match industries for coverage |
| Customer Preferences or Standards | External Certifications / Internal Certifications | Keyword Matching / Set Overlap | Standards/compliance mapping. Cleaning and tokenizing strings into sets, then measuring overlapping of keywords. |
| Products Involved | Product Experience | Category Similarity Coverage (Fuzzy match with threshold) | Matching Products for coverage |
| Integration Requirements | Expertise | Keyword Matching / Set Overlap | Technical integration alignment. Cleaning and tokenizing strings into sets, then measuring overlapping of keywords. |
| Required Skills and Expertise | Core Competencies | Category Similarity Coverage (Fuzzy match with threshold) | Primary skill matching coverage |
| Complexity Rating | Core Competencies (expertise level) | Coverage x expertise fit (employee capability score) then compare complexity | Compare Employee Capability based on skills to Complexity |
| Work Location | Work Location | Category Similarity (Fuzzy match with threshold). Irrelevant if remote | Location coverage |
| Work flexibility | Work flexibility | Category Similarity (Fuzzy match with threshold) | Remote/Hybrid/On-site scoring factoring for location coverage |
| Language Requirements | Language Proficiency | Category Similarity Coverage (Fuzzy match with threshold) | Check if required language is present for coverage |
| Language Level | Communication skills | Coverage x language fit (language capability) | Scoring using match or near-match on required fluency factoring for language coverage |
| Effort | Weekly Availability in Hours | Filter on availability after scoring | Filter on availability after scoring |
| Requested End | Available From | Filter on availability after scoring | Filter on availability after scoring |
| - | Cultural Awareness | Optional Bonus (Rating Match) | Use as soft bonus if needed |
| - | Collaboration | Optional Bonus (Rating Match) | Soft skill scoring if needed |
| - | Problem Solving | Optional Bonus (Normalized Score) | Tiebreaker or bonus |
| - | Leadership | Optional Bonus (Experience-Based) | Soft bonus for senior roles |

**Matching Different Features:**

**Product Match Score**

**Tables and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Products Involved | Employees | Products Experience |

Here we will have a list of products in the projects table and similarly a list in the employees table. As the products involved field in projects data is human filled, we first use fuzzy matching to get rid of any mismatching with products experience column in employee table caused by spelling mistakes and then we match the columns. The coverage is calculated based on how many required products are possessed by the employee. For example, if a project requires products AIScan, Print2.0 and Workflow2000 and employee knows only AIScan, the coverage would be 33%. If an employee knows any two of the above then 66% and in case of knowing all three or more, it would be 100%.

**Location Match Score**

**Table and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Work Location | Employees | Work Location |
| Projects | Work Flexibility | Employees | Work Flexibility |

Scoring Mechanism:

This calculation assigns a **location match score** based on both:

1. **Work flexibility compatibility**
2. **Location similarity (using fuzzy matching)**

Note: As Locations are filled by Humans, to match them we would use fuzzy matching.

| **Project Flexibility** | **Employee Flexibility** | **Location Match** | **Score** | **Explanation** |
| --- | --- | --- | --- | --- |
| remote | Any | — | 1.0 | Location irrelevant for remote work |
| Any except remote | Any | No | 0.0 | Location mismatch or wrong flexibility |
| onsite | onsite | Yes | 1.0 | Perfect location and presence match |
| onsite | hybrid | Yes | 0.5 | Partial match; available some days onsite |
| onsite | remote | Yes | 0.0 | No physical presence at required location |
| hybrid | onsite | Yes | 1.0 | Can fully accommodate onsite days |
| hybrid | hybrid | Yes | 1.0 | Flex on both sides; good match |
| hybrid | remote | Yes | 0.5 | Remote match possible but less ideal |

**Language Match Score**

**Language Match with Proficiency Logic**

We match project language requirements with employee language fluency, handling:

* Typos/misspellings in language names (fuzzy matching)
* Fluency level comparison using the CEFR scale
* Finally, scoring based on coverage and fluency fit

**Table and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Language Requirements | Employees | Language Proficiency |
| Projects | Language Level | Employees | Communication skills |

**Mapping used for CEFR Scale:**

**CEFR Level Numeric Value**

**A1 1**

**A2 2**

**B1 3**

**B2 4**

**C1 5**

**C2 6**

**Steps for calculating Language Score:**

1️. For each project language, find the best fuzzy match in an employee’s known languages

2️. If a match is found:

Compare CEFR levels using the cefr\_scale

3️. Score per language:

• If employee level ≥ required → score = 1.0

• Else → 1 - (diff / 6)

4.Calculate Average Fit:

The mean of all individual language scores per language (fluency comparison), **only** for matched languages.

5. Coverage = matched languages / total required

6. Final Score = coverage × average fit

**Industry Match Score**

**Tables and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Customer Industry | Employees | Industry Experience |

Here we will have a customer industry column in the projects table containing a category based on the industry of the client. Similarly, in the employee’s table we would have an Industry Experience column containing a list of different industries the employee is experienced in. As the industries in the projects data is human filled, we first use fuzzy matching to get rid of any mismatching with industry experience column in employee table caused by spelling mistakes and then we match the columns. The score would be 1 if the employee is experienced in the client industry and 0 otherwise. For example, if a project is from a “retail” clients and employee has experience with “Manufacturing”, “Education”, “Retail”, the score would be 1. If an employee knows “Manufacturing” and “Education” or any list excluding “Retail” the score would be 0.

**Skills Match Score**

**Table and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Required Skills | Employees | Core Competency (key) (Skill name) |
| Projects | Complexity Rating | Employees | Core Competency (value) (Skill expertise rating) |

Scoring is done on how well an employee's core competencies align with the skills required for a project, considering:

1. **Coverage** — How many required skills they know.
2. **Expertise Fit** — How experienced are they in those skills.
3. **Complexity Fit** — How well their skills meet the challenge of the project

**Steps for calculating Skill Score:**

1. **Fuzzy Skill Matching**  
   Match each required skill (may contain typos) with the closest skill in the employee’s core competency using fuzzy logic.
2. **Coverage Score**  
   Calculate what fraction of required skills are present in the employee’s skill set.  
   **Formula:**  
   coverage = matched\_skills / total\_required\_skills
3. **Expertise Fit**  
   For **each matched skill**, compare employee’s level to the required level:
   * If level is equal or higher → score = 1.0
   * If lower → score = 1 - (required - actual) / 10
   * Average these for **expertise\_fit**.
4. **Capability Score**  
   Multiply **coverage** by **expertise\_fit**  
   capability = coverage × expertise\_fit
5. **Final Skill Score (Complexity Fit)**  
   Compare capability with project complexity rating (normalized as complexity / 10):
   * If capability ≥ complexity → score = 1.0
   * Else → capability / complexity

**Certification Match Score**

**Tables and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Customer Preferences (Certifications) | Employees | External/Internal Certifications |

Here we will have a list of certifications in the projects table that the client demands are must have. Similarly, a list of certifications that the employee has completed would be in the employees table. As the certifications field in projects data is human filled, we first use fuzzy matching to get rid of any mismatching with certifications column in employee table caused by spelling mistakes and then we match the columns. The coverage is calculated based on how many required certifications are possessed by the employee. For example, if a project requires Microsoft Azure Certification and ISO 27001 and employee has only ISO 27001, the coverage would be 50%. If an employee knows both above (or additional) then 100%. If no certifications are required, the employee gets a full score (100%).

**Expertise Match Score**

**Tables and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Integration Requirements (Expertise Areas) | Employees | Expertise Areas |

Here we will have a list of expertise areas required for the project in the projects table Similarly, a list of employee expertise in the employee’s table. As this field in projects data is human filled, we first use fuzzy matching to get rid of any mismatching with expertise column in employee table caused by spelling mistakes and then we match the columns. The score is calculated based on the coverage of required expertise against possessed expertise, identical to the certification score calculation.

**Job Description Match Score**

**Table and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Project Summary | Employees | Role Description |
| Projects | Scope and Deliverables | Employees | Role Description |

To evaluate how well an employee’s role matches the intent of a project, we compare the **Project Summary** and **Scope and Deliverables** fields with the employee’s **Role Description**. We use **TF-IDF vectorization with cosine similarity** to compute the semantic closeness between the text pairs.

The two similarity scores (one for the summary, one for the scope) are **averaged** to produce a final Job description match score, representing how relevant an employee’s responsibilities are to the project described.

| **Comparison Pair** | **Description** | **Output** |
| --- | --- | --- |
| Project Summary vs Role Description | Measures how well the overall project aligns with the role | summary\_sim (0–1) |
| Scope and Deliverables vs Role Description | Measures how well the specific responsibilities align | scope\_sim (0–1) |
| **Final Job Description Score** | Average of the two scores | (summary + scope) / 2 |

***Note on TF-IDF:****Term Frequency – Inverse Document Frequency emphasizes meaningful words and down weights common ones (like “and”, “the”, etc.). This allows us to represent entire sentences as vectors and measure their cosine similarity, which reflects semantic alignment.*

**Availability Filtering**

**Tables and Columns Used**

| **Table 1** | **Variable** | **Table 2** | **Variable** |
| --- | --- | --- | --- |
| Projects | Effort | Employees | Weekly Availability in Hours |
| Projects | Requested End | Employees | Available From |

The **Availability Score** measures whether an employee can complete a project within the required time frame based on their availability start date and weekly working capacity.

**Steps For scoring:**

1. **Compare dates**: First we compare employee’s Available From date to project’s Requested End date. If the employee is available only after Requested End Date then the score is 0.0 (not available).
2. **Estimate working time**:
   * Calculate total **calendar days** between employee’s Available From Date and Project’s Requested End Date.
   * We convert the calendar days to **working days** using a 5/7 multiplier (for a standard 5-day workweek).
   * Derive **available working weeks** by dividing working days by 5.
3. **Calculate required effort per week**:
   * Required Weekly Effort = Project Effort / Available Weeks
4. **Score logic**:
   * If Required Weekly Effort ≤ Employee Weekly Capacity, then we say the score is 1.0
   * Otherwise, the score is a decimal ratio (e.g. 0.7), based on how much of the requirement the employee can handle.

Additionally, an **Available** flag is created that is 1 if Availability Score is 1 and 0 otherwise. The reason to create “Availability Score” and “Available” flag separately are that if no instance has Available Flag as 1, we can consider high Availability Score employees for the project.

**Final Ranking**

To calculate **the final ranking of employees for each project**, we followed a structured multi-step approach:

1. We filtered the dataset to include only those employees marked as available (Available == 1). This flag is explained above. Filtering on this flag ensures that ranking and recommendation were limited to candidates that had the bandwidth to take up and complete the project within the desired timeline.
2. Next, we defined a set of adjustable weights for each of the score dimensions discussed above, such as job description match, product match, location compatibility, skill alignment etc. These weights reflect the relative importance of each criterion in determining a good project–employee fit.
3. Using these weights, we then computed a weighted score for each employee–project pair by multiplying each individual score with its corresponding weight and summing the results. This resulted in a **comprehensive Final Score** representing the overall suitability of each employee for a given project.
4. After computing the final scores, we sorted the data by ProjectID and ranked the employees in descending order of their weighted scores. This gives the ranking of each employee based on project criteria.

Optionally, we can also add other sorting or filtering criteria such as availability score or location matching before sorting on weighted scores. This will allow extra flexibility in extracting top employees if we need strict location matching or more relaxed availability matching.

This weighted scoring and ranking process ensured that employee recommendations were both quantitative and context-aware, based on a configurable combination of hard skills, logistics, experience, and soft skills.