TECHNISCHE UNIVERSITÄT WIEN

194.207 Generative AI (VU 4,0) 2025W

Project: SkillSync

Authors: Group 45
Shahzad Muhammad Azeem (12346021),
Lasheen Nooreldin (12302427),
Baranga Roxana Mary (12502784),
Kormaku Ana (12534172),
Şaban Akay (12045645)

Vienna, December 10, 2025

1 Project Overview

The goal of this project is to automatically form balanced, diverse, and complementary teams of students or employees by analysing skills, experience, and preferences extracted from profile inputs. Instead of relying on chance or incomplete information, the system uses Generative AI and algorithmic assignment to create groups that are more effective, fair, and aligned with the tasks they need to complete.

This project was inspired by the experience of seeing groups assigned randomly in many university courses. While randomness is quick, it often leads to unbalanced teams where several people share the same strengths, while other important roles or skills are missing. This results in frustration, uneven workload distribution, and missed learning opportunities. Similar challenges appear in companies, organisations, and agile teams, where assembling well-rounded groups is essential for productivity and innovation.

To address this, the system employs a Large Language Model to extract structured, machine-readable information from short profile forms. The extracted data includes technical skills, soft skills, preferred roles, past experience, and other relevant attributes. These skills are then normalised and transformed into numerical representations so they can be compared objectively across candidates. Using this information, optimization and matching algorithms determine how to combine individuals into teams that are complementary in skills, balanced in capability, and similarly aligned with overall project requirements.

By integrating AI-powered skill extraction with a structured optimization engine, the project offers a scalable and repeatable method for building effective teams in educational settings, companies, organisations, and any environment where teamwork is essential.

2 User Group, Goals, and Workflows

2.1 User Group

The system targets Project Managers (PMs), Team Leads, and Coordinators in medium-to-large engineering, technology, and research organizations. This also directly connects to the academic context of TU Wien, where students and researchers frequently form project teams and collaborations in courses, labs, and research groups. The chosen domain reflects real workflows many students (including our group) regularly experience during group-based university projects, hackathons, and research collaborations.

2.2 User Goals

Users aim to:

- Build balanced teams that cover all required competencies.
- Identify complementary skills and avoid unnecessary overlap.
- Match working styles and availability to ensure smooth coordination.
- Make data-informed, unbiased decisions using structured information.
- Reduce the time spent manually searching for and interpreting fragmented data.

3 User Data and Information Sources

3.1 Data Types

SkillSync uses a combined dataset sourced from public datasets, semi-structured documents, and a small amount of synthetic data. Primary sources include Kaggle's Resume Dataset.
The collected data includes:

- Surveys (CSV)
- Metadata fields include timestamps, department, seniority, past project roles, industry, collaboration style, AI-interaction habits, frustration points, education level, and country information.

3.2 Target Dataset and Size

We will extract a subset of **200 anonymized user profiles** that match our standardized schema and project requirements.
Each profile includes:

- 5–10 technical and soft skills
- 1–3 short project or experience summaries
- Standardised metadata fields such as role, seniority, industry, availability, collaboration style, AI-interaction attributes and last active.

Size justification:

- Sufficient for evaluating skill extraction precision/recall
- Large enough for retrieval and ranking benchmarking
- Provides enough diversity to test team compatibility modeling
- Manageable to curate and label within course workload constraints

3.3 Data Characteristics Across Five Dimensions

3.3.1 Granularity

The dataset contains information at varying levels of granularity. Some elements are highly atomic, such as individual bullet-point skills, while others are embedded within longer narrative descriptions. Skills and competencies often appear implicitly inside paragraphs, requiring extraction through natural language processing techniques.

3.3.2 Connections

Relationships between skills, past project roles, collaborators, and work history are not explicitly encoded. These connections must instead be inferred using semantic embeddings or contextual similarity models.

3.3.3 Completeness

Profile completeness varies considerably. Some records are well-structured and detailed, while others lack essential fields such as soft skills, availability information, or project summaries. The synthetic dataset intentionally incorporates varying degrees of completeness to support robust model evaluation.

3.3.4 Context

Contextual metadata, including timestamps, project relevance, and seniority progression, is inconsistently available across profiles. When such metadata is missing, the system must approximate or infer contextual information.

3.3.5 Heterogeneity

The profiles exhibit substantial variation in formatting, structural organization, writing style, tone, and length. This heterogeneity is expected and forms a central challenge for consistent parsing and downstream processing.

4 Problem

Modern team formation suffers from fragmented, inconsistent, and inaccessible information. Project Managers and coordinators must make critical assignment decisions without a holistic view of individuals' skills, experience, or collaborative potential. Essential information is often unstructured, scattered across documents and platforms, implicit rather than explicit, and difficult to retrieve or synthesize. As a result, teams are frequently assembled based on incomplete insights rather than data-driven reasoning.
This leads to recurring operational challenges: missing combined competencies, hidden relationships that remain undiscovered, difficulty synthesizing how people fit together, and overall knowledge fragmentation across PDFs, emails, notes, and informal channels.

4.1 Limitations of Traditional Team-Formation Process

The existing workflow prevents reliable team formation due to structural limitations:

- **Fragmented information:** data scattered across multiple unconnected sources.
- **Incomplete soft-skill and availability data:** crucial attributes missing or hidden.
- **Subjective and slow profile comparison:** no scalable mechanism to infer patterns.
- **Heterogeneous formats:** inconsistent formatting complicates automated processing.
- **Bias-prone decisions:** reliance on familiarity or partial information leads to unbalanced teams.

4.2 Objective

SkillSync addresses these challenges by creating a unified, AI-driven approach to understanding individual profiles, identifying complementary skill sets, uncovering collaboration patterns, and recommending balanced teams tailored to project requirements. The system produces clear, human-readable explanations to support transparency and trust.

5 Solution concept

High-Level System Tasks SkillSync performs 4 core tasks:

- **Understand:** Extract skills, roles, experience, and contextual cues from raw documents.
- **Retrieve:** Identify suitable candidates for specific roles, queries, or project needs.
- **Connect:** Discover semantic relationships, complementarities, and hidden collaboration patterns.
- **Generate:** Produce summaries, rationales, and explanation narratives supporting recommendations.

User Input → Document Understanding → Embedding & Matching → Team Recommendation → Explanation Generation

6 Technical approach

Our system follows a multi-stage workflow that integrates traditional data processing, algorithmic decision-making, and LLM-assisted reasoning. The pipeline consists of three major components: data preprocessing, information extraction, team construction and LLM-based explanation.

6.1 Preprocessing Layer

The first stage prepares all raw user data before any semantic interpretation occurs. This step focuses purely on cleaning and standardizing inputs:

- text cleaning (removing noise, punctuation inconsistencies, redundant tokens),
- normalization of multi-value fields (skills, tools, collaboration style),
- standardization of terminologies (e.g., "JS" → "JavaScript"),
- splitting comma-separated lists and enforcing consistent formatting,
- handling missing values in structured fields.

The output of this layer is a cleaned dataset where all fields follow a consistent format, making them ready for machine-assisted interpretation.

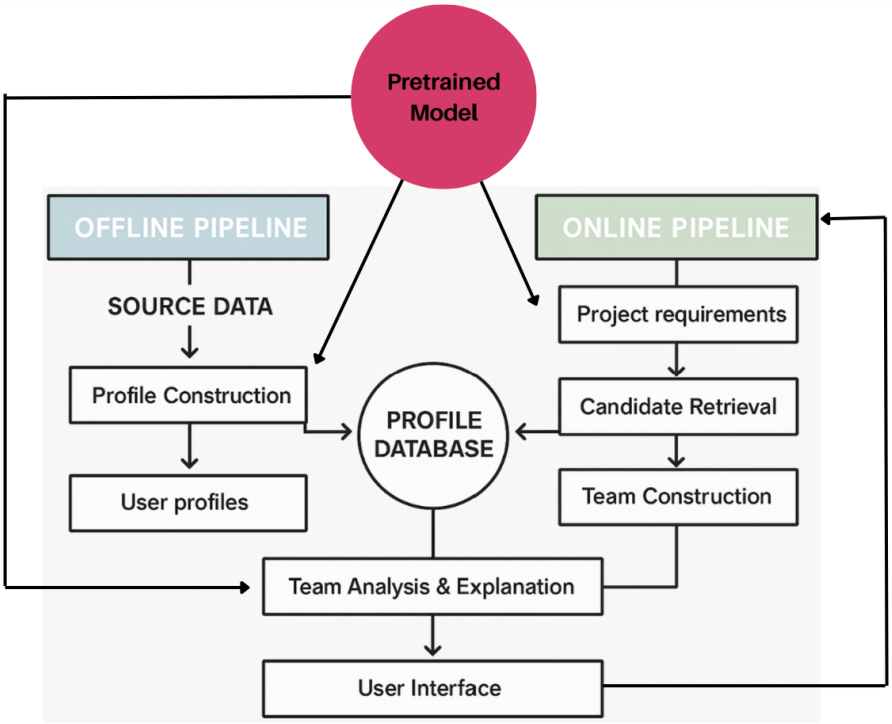

Figure 1: System architecture

6.2 Information Extraction Layer - LLM

In the second stage, the system applies an LLM to convert free-text fields from both candidate's profiles and project description into structured, machine-interpretable attributes. The output for each candidate is a standardized JSON profile that captures all relevant semantic information in a consistent schema. From each candidate, we extract:

- skills and tools,
- preferred roles,
- experience or domain indicators.

The output for the project description is a structured requirement object:

- required roles,
- team sizes,
- mandatory skills.

This layer ensures that heterogeneous inputs are transformed into a uniform profile suitable for downstream semantic processing.

6.3 Embedding Layer - Semantic Vectorization

After the LLM has produced structured profiles, a separate embedding model converts each profile into a fixed-length vector representation. These embeddings are used to compute:

- candidate-candidate similarity and team cohesion,
- candidate-task fit.

The embedding model provides numerical vectors that allow reliable evaluation. This layer is critical for ranking, matching and optimization during team construction.

6.4 Team Construction Layer - Algorithmic Approach

In this stage, the app forms teams using the structured candidate profiles and semantic embedding vectors generated in the previous layers.
After this, each team should have:

- combined coverage of required skills

6.5 LLM-Assisted Team Explanation Layer

Teams are passed to the LLM for interpretive evaluation. The model generates human-readable explanations describing:

- why the team matches the project needs,
- its strengths and weaknesses,
- potential risks or gaps, Explanations must be consistent with retrieved data and avoid hallucinations.

7 User Interface

The system will be implemented as a lightweight web application (Streamlit-based prototype). The goal of the interface is to make team formation transparent and easy to iterate on, without requiring installation or technical expertise.

- **Upload profile data** (CSV following our standardized schema),
- **Enter project requirements** using natural language and optional structured fields,
- **Inspect recommended teams** and their explanations through a clear results interface.
- **Candidate distributions** (skills, availability, roles),

8 Evaluation

The goal of the evaluation is to verify whether the system functions correctly at each processing layer and that the generated teams are meaningfully better than random assignment. We evaluate six core components:

- profile extraction accuracy
- requirement interpretation
- team construction quality
- explanation quality
- end-to-end latency

8.1 Profile Extraction Accuracy

Procedure: Compare LLM-generated profiles with 15–20 manually annotated ground-truth profiles.
Metrics:

- Field accuracy (correct fields / total fields),
- Precision and recall for extracted skills,
- JSON schema validity (structure, keys, formatting).

8.2 Requirement Interpretation

Procedure: Evaluate the LLM on 5–7 project descriptions with predefined ground-truth requirement objects.
Metrics:

- Correct extraction of required roles and team size,
- Low hallucination rate (invented skills or constraints).

8.3 Team Construction Quality

Procedure: For 10 team-generation scenarios (different subsets of candidates and project descriptions), we compare the teams produced by our system to randomly generated teams that respect the same role counts and team sizes.
Metrics:

- Constraint satisfaction: percentage of teams that exactly match the required role composition (target: 100%),
- Task fit: average cosine similarity between team members and the project embedding; system teams should score higher than random teams,
- (Optional) Cohesion: average cosine similarity between members of the same team.

8.4 Explanation Quality

Procedure: Review 10–15 generated explanations and compare them to the structured profiles.
Metrics:

- Clarity and coherence,
- Consistency with actual candidate data,
- Absence of hallucinated skills or experience.

8.5 End-to-End Latency

Procedure: Measure system performance using datasets of 15, 25, and 50 profiles.
Metrics:

- Total pipeline latency,
- LLM inference time,

9 Tech Stack

SkillSync will be implemented using a lightweight and modular technology stack that supports LLM-based extraction, semantic evaluation, and team construction.

9.1 Backend

- Python 3.x for preprocessing, profile handling, and team-formation logic.

9.2 Models

- OpenAI GPT models for skill extraction, requirement interpretation, and explanation generation.
- Sentence-Transformers (e.g., all-MiniLM-L6-v2) for embedding and similarity scoring.

9.3 Storage & Retrieval

- FAISS or similar vector index for efficient similarity search.
- CSV/JSON files for storing cleaned profiles and structured representations.

9.4 Algorithms

- NumPy/SciPy for numerical operations and scoring.
- Custom heuristic/constraint-based methods for assembling balanced teams.

9.5 Frontend

- Streamlit for an accessible, browser-based interface for data upload, requirements entry, and viewing team recommendations.

10 Timeline

10.1 Week 1 - Preprocessing

Collect dataset, define the final JSON schema for candidate profiles and project descriptions, implement preprocessing pipeline and define evaluation ground truth criteria.

10.2 Week 2 - Extraction

Create prompt design for candidate profile extraction and project requirements extraction, implement structured extraction for JSON and evaluate extraction accuracy.

10.3 Week 3 - Embedding

Implement embedding generation for candidates and project descriptions, implement cosine similarity scoring functions, evaluate semantic distances on test data.

10.4 Week 4 - Team Construction and Explanation

Implement assignment algorithm using embeddings, enforce role constraints, implement project-fit scoring, build the LLM-based explanation generator, evaluate team quality against random baseline.

10.5 Week 5 - Finalize Project

Build Streamlit interface, integrate all layers into the pipeline, measure latency on multiple profiles, complete final quality evaluation, prepare presentation.

10.6 Week 6 - Backup

Backup week, for unforeseen circumstances.

11 Summary

Our automates team formation by combining LLM-powered data extraction with embedding-based semantic scoring and an assignment algorithm. The LLM converts both candidate profiles and project descriptions into structured data. After this, a separate embedding model provides numerical vectors for evaluating candidate-project fit. Using these, the system assigns candidates to the required project roles, ensuring that each team satisfies the specified structure. Finally, the LLM generates concise explanations describing why each candidate was selected.