International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 11 | Nov 2026 www.irjet.net p-ISSN: 2395-0072

INTERNAI: AN AI-BASED INTERNSHIP RECOMMENDATION AND CAREER GUIDANCE SYSTEM

C. Sanjana Reddy[1], D. Sahasra[2], D. Sri Nithya[3], J. Tejashwini[4]

[1234]*Department of Computer Science and Engineering, Hyderabad Institute of Technology and Management, Hyderabad, Telangana, India*

Abstract – *Students searching for internships face a very large number of listings and almost no help in judging which ones fit them. This paper presents InternAI, a web-based recommender that compares a student's profile with internship requirements and returns a ranked list in which every entry carries a match percentage, the skills already held, the skills still missing, and a short written reason. Matching is a weighted sum of four components, namely skill coverage, domain agreement, location compatibility and academic eligibility, with weights of 50, 25, 15 and 10 percent. The first-phase prototype is written in HTML5, CSS3 and JavaScript, runsentirely in the browser and is hosted on netlify. Functional, matching, ranking, skill-gap, interface and end-to-end tests were all passed on a sample dataset, and a worked example shows how a score is obtained. The paper also states the limits of this phase, including exact-string skill comparison and a small static dataset, and outlines a path toward NLP-based skill extraction and TF-IDF similarity.*

Keywords: Intership recommendation, weighted matching, skill gap analysis, Explainable recommendation, Career guidance, Content-based filtering.

1. INTRODUCTION

Students now come across internship openings on job portals, institutional notice boards, messaging groups and company career pages. The abundance does not lead to better decisions. Every posting has to be read, interpreted against the applicant's own abilities and compared with many others, and most portals support this only through keyword serach and filters. Such tools retrieve postings, but they say little about how well a particular applicant fits a particular posting.

This paper describes InternAI, a system that takes over part of that comparison work. A student submits a short profile made up of skills, a preferred domain, a preferred location and an academic background. The system scores every internship in its collection against the profile, orders the results, and attaches to each one the skills the student already hold, the skills that are still lacking, and one-line justification. The intended shift is from the questin "which intership should I choose?" to the question "why does this one suit me, and what should I learn next?".

The project proceeds in phases, and this paper reports the first. Its contributions are: (i) a transparent weighted-matching model over four profile features; (ii) a skill-gap guidance; (iii) a serverless prototype that is publicly deployed; and (iv) a structured test of the complete flow from profile entry to ranked output. Naturanl language processing (NLP) based skill extraction, Term Frequency-Inverse Document Frequency (TF-IDF) similarity, persistent storage and live data are intentionally left for later phases.

Section 2 revies related technique, Section 3 presents the design and the scoring model, Section 4 covers the implementation, section 5 discusses results and limitations, and section 6 concludes.

2. RELATED WORK

A recommender system narrows large collection of items to those likely to interest a particular use. Content-based filtering describes users and items through attributes and proposes items whose attributes resemble the user's preferences [1]. Since it needs no ratings or behavior from other users, it suits a platform that has no usage history yet, which is the position of InternAI. In our setting the item attributes are required skills, domain, location and eligible academic backgrounds.

When items are described by long free text, the text can be turned into weighted term vectors with TF-IDF and compared with the profile using cosine similarity. This phase does not use it; the prototype checks only how many of the required skills the student holds. Hybrid designs combine several recommendation techniques, but they work best when there is enough data on what users liked or applied for, which a new system does not have.

A Student project at BMs Institute of Technology and Management has built an AI-based internship recommendation engine around student profiles, showing that profile matching saves a student the effort of searching manually [2]. Research on explainable AI further points out that users trust a recommendation more when the system explains the reason behind it [3]. InternAI follows this direction but keeps the model deliberately simple, so that its score can be recomputed by hand and its reason can be read directly from the score components.

3. SYSTEM DESIGN

3.1 Architecture

InternAI has three layers (Fig. 1). The user layer collects the profile. The recommendation layer normalizes the entered skills, computes the weighted score for each explanation. The output layer presents the match percentage, the ranked list, the skill gap and the reason.

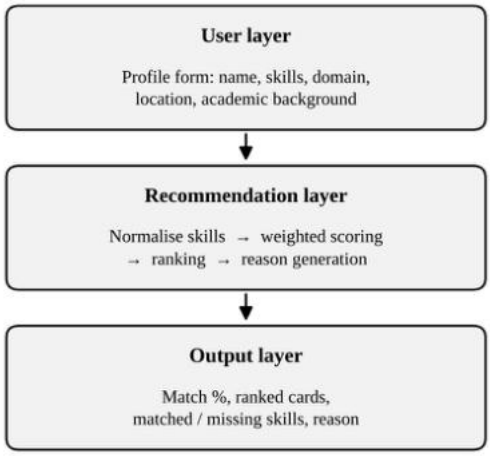

Fig -1: Three-layer architecture of InternAI

3.2 Input Data and Dataset

The student supplies five inputs: name, skills, preferred domain, preferred location and academic background. The domain is chosen from five options (Software Development, Web Development, Data science, Artificial Intelligence, Cybersecurity) and the location from four (Hyderabad, Bangalore, Chennai, Remote). Each record of the sample dataset holds an internship title, a domain, a location, a list of required skills and the academic backgrounds for which it suitable, The records are kept as JavaScript objects, so no server is required.

3.3 Scoring Model

Let S be the set of skills held by the student and R the set required by an internship. The skill score is the fraction of requited skills that the student already has:

$$S_s = |S \cap R| / |R| \quad (1)$$

Three further scores are binary. The domain score S_d is 1 when the preferred domain equals the internship's domain. The location score S is 1 When the preferred location equals the internship's location or the internship is remote. The

$$M = 100\,(w_s S_s + w_d S_i + w_l S_i + w_a S_a) \quad (2)$$

$$w_s + w_d + w_l + w_a = 1 \quad (3)$$

Skills receive the largest weight because they decide whether a student can actually do the work (Table 1). Because the weights sum to one and every component lies between 0 and 1, Malways lies between 0 and 100, and a score of 100 requires full skill coverage together with agreement on domain, location and academic eligibility. A consequence of the design is that a remote internship always earns the full location share, and an eligible degree alone earns 10 points even when nothing else matches.

Table -1: Weights of the scoring components

Component	Symbol	Weight	Score type
Skills	ws	0.50	Fraction of required skills held
Preferred domain	wd	0.25	Binary (1 or 0)
Preferred location	wl	0.15	Binary (1 or 0)
Academic background	wa	0.10	Binary (1 or 0)

3.4 Ranking and Explanation

After all internships are scored they are sorted in descending order of M. For each one the system derives the matched skills (S ∩ R) and the missing skills (R − S), whose combination is the skill gap. A short reason is then assembled from the components that contributed to the score, for example a domain match, a location match, and the number of required skills already held.

4. IMPLEMENTATION

4.1 Frontend

The interface is built with HTML5 and CSS3. A single form holds the five fields of Section 3.2 and a *find Intership*s button that starts the matching. Results appear as cards, each showing the match percentage, the matched and missing skills and the reason.

4.2 Matching Logic

The logic is writtern in JavaScript and runs in the browser. On submission the entered skills are cleaned by removing extra spaces and unifying letter case, and are soted as a set. The procedure of Algorithm 1 is then applied to every record of the dataset.

Algorithm 1. Match and rank internships
Input: skills *S*, domain, location, academic background; dataset *D*1.
Normalise entered skills (trim, unify case) and store in *S*.

2. **for each** internship in *D*, with required skills *R*:
3. $Ss = |S \cap R| / |R|$
4. set *S*d, *S*l, *S*a to 1 or 0 by the rules of Sec. 3.3
5. $M = 100 \times (ws.Ss + wd.Sd + wl.Sl + wa.Sa)$
6. matched = $S \cap R$; missing = $R - S$; build reason text
7. Sort internships by *M* in descending order.

8. Render each result as a card.

4.3 Development and Depolyment

The code was written in Visual Studio Code and previewed with the Live Server extension. After testing, the prototype was deployed on Netlift, which provides a public link that opens in any modern browser. Development needs only modest hardware: an IntelCore i3 class processor, 4 GB of RAM (8 GB recommended) and about 500 MB of free disk space.

4.4 Test Method

Six kinds of tests were carried out. Functional tests exercised the input fields, buttons and navigation. Matching tests checked that given skills and preferences produce the expected score. Recommendation tests checked that results are ordered by match percentage. Skill-gap tests checked the matched and missing skills of each result. Interface tests checked the layout and cards, and an integration test followed the complete path from profile entry to ranked output.

5. RESULTS AND DISCUSSION

5.1 Worked Example

Consider a student who knows Python, SQL and Statistics, prefers Data science in Hyderabad, and has a Computer Science and Engineering background. Table 2 lists the scores for three illustrative internships; their names and requirements are examples, not entries quoted from the dataset. The Data Analyst internship (Data Science, Hyderabad) requires Python SQL, Excel and Statistics. The ML internship (Artificial Intelligence, Bangalore) requires Python, ML and TensorFlow. The Web Developer internship (Web Development, Remote) requires HTML, CSS and JavaScript.

Table-2: Illustrative mathc scores

Internship	Ss	Sd	Sl	Sa	M (%)
Data Analyst	0.75	1	1	1	87.5
ML Intern	0.33	0	0	1	26.7
Web Developer	0.00	0	1	1	25.0

For the first internship, M = 100 × (0.50 × 0.75 + 0.25 × 1 + 0.15 × 1 + 0.10 × 1) = 87.5 %. Only Excel is missing, so the student can see directly that learning it would close the gap. The ranking is therefore 87.5 %, 26.7 % and 25.0 %. Fig. 2 splits each score into the contribution of its four components, which is the information the generated reason is built from.

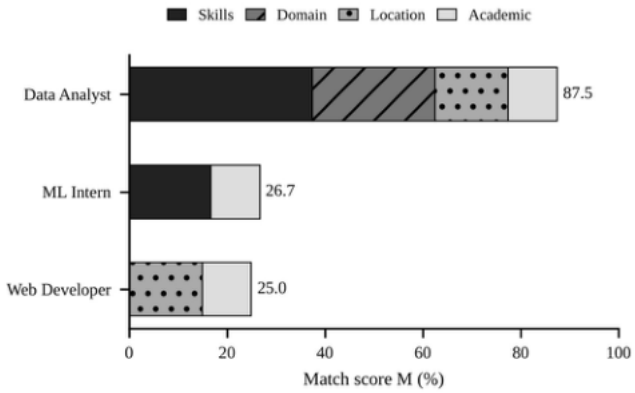

Fig -2: Contribution of each component to the match score

5.2 Test Outcome

All six test categories were verified on the prototype (Table 3). These tests establish that the implementation behaves as specified; they are not a measurement of recommendation accuracy.

Table -3: Summary of testing

Test	What was verified	Outcome
Functional	Inputs, buttons, navigation	Verified
Matching	Expected scores from given inputs	Verified
Recommendatio	Ranking by match	Verified
Skill gap	Matched and missing skills	Verified
Interface	Layout and card display	Verified
Integration	Profile entry to ranked	Verified

5.3 Comparison with Manual Search

Table 4 contrasts the prototype with the manual search that students currently perform. The gain is mainly in effort and transparency: one form yields an instant ranked list, a numeric relevance measure, a visible skill gap and a stated reason.

Table -4: Manual search versus InternAI (phase 1)

Aspect	Manual search	InternAI
Effort	Read and compare many listings	One form, instant ranked list

Relevance	Student's own judgement	Percentage match score
Skill gap	Not shown	Matched and missing skills
Reason	Not given	Shown with every

5.4 Limitations

The prototype has several limits. The dataset is small and static. Skills are compared exactly as typed, so "JS" and "JavaScript" count as different skills, which understates the skill score of students who use abbreviations. The weights are fixed and not learned from feedback, the three binary components are coarse, and nothing is stored between sessions because there is no backend. No user study has yet been conducted, so the claim that the recommendations are useful rests on the design and the functional tests rather than on measured user outcomes.

6. COCLUSIONS AND FUTURE WORK

InternAI shows that a simple, inspectable weighted model can turn a student profile into a ranked list of internships with matched skills, missing skills and a written reason. The browser-only prototype is deployed on Netlify and passed all tests planned for this phase. Because scoring is a separate step, it can later be replaced without changing the interface.

Future phases will add the following:

- NLP-based skill extraction from resumes and internship descriptions, with synonym handling so that equivalent skills are treated alike;
- TF-IDF weighting with cosine similarity for matching against descriptive text;
- A larger dataset with a backend and database for accounts and saved results, fed by live internship data;
- Pointers to learning resources for missing skills, and a user study to measure recommendation quality.

ACKNOWLEDGEMENT

The authors thank the Program Head, Dr. Dharmendra K. Roy, and the PBL-I coordinator, Mrs. Sure Mamatha, of the Department of Computer Science and Engineering for their guidance.

REFERENCES

[1] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in The Adaptive Web, Lecture Notes in Computer Science, vol. 4321, Springer, 2007, pp. 325–341.

[2] "AI Based Internship Recommendation Engine," project work, BMS Institute of Technology and Management.

[3] Research papers on Explainable AI (as referred to in the project review presentation).