International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

InternReady: Student Internship Skill Gap Analyzer

Sandhya, G. Ravi Priya, G. Prem Sagar, G. Varun, K. Jaya Krishna

Roll numbers: 25E51A0592, 25E51A0596, 25E51A0507, 25E51A05A8, 25E51A0512

Guide: Mrs. Jahanara Begum

Department of Computer Science and Engineering, Hyderabad Institute of Technology and Management (HITAM), Hyderabad, India

Abstract. Internships give students practical industry experience, but many students do not know which skills a particular internship role expects, and at what depth. Role requirements are scattered across job portals, learning platforms and assessment tools, so students compare them manually, which is slow and confusing. This paper presents InternReady, a web-based skill-gap analyzer. A student creates a profile (name, branch, year), selects a target internship role, and rates each required skill on a four-level scale (0 Not Learned, 1 Beginner, 2 Intermediate, 3 Advanced). The system compares the self-rated levels with the role's required levels, classifies every skill as meeting the requirement, weak or missing, computes a readiness score, and generates an ordered improvement roadmap together with a downloadable report. The architecture has four layers: an HTML/CSS presentation layer, a Python application layer containing the analysis engine, and a MySQL data layer holding students, roles, skills and role requirements. This version of the paper specifies the methodology in full: the data model, the skill-level scale, the matching and gap-classification rules, the scoring formula, the roadmap-ordering rule and the test plan, and it includes the architecture, workflow and database diagrams together with screenshots of the working prototype. The readiness figures shown in the screenshots are sample outputs of the prototype; no measured accuracy or user-study results are claimed, and the planned tests are listed as planned.

Keywords: InternReady, Internship Preparation, Skill-Gap Analysis, Readiness Score, Learning Roadmap, Python, HTML, CSS, MySQL, Student Employability

1 Introduction

Internships connect classroom learning with industry practice and are often a student's first step towards employment. A student who applies for a role such as Full Stack Developer Intern, however, is usually unsure which skills are expected (for example HTML/CSS, JavaScript, React, SQL and Git), how deep the knowledge should be, and which gap to close first. The information exists, but it is spread across job portals that list requirements, learning platforms that offer courses, and assessment tools that measure ability. Connecting these three sources is left to the student.

InternReady removes this manual step. It stores the required skill levels for each supported internship role, asks the student to rate their own level in each of those skills, and returns a structured comparison: which skills already meet the requirement, which are weak, which are missing, how ready the student is overall, and in what order the gaps should be addressed. The tool is meant for self-assessment and planning and does not replace formal assessment or the selection criteria of any employer.

1.1 Objectives

- Let a student create a skill profile and select a target internship role through a simple web interface.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

- Represent role requirements and student skills on a common four-level scale so that they can be compared numerically.
- Identify, for every required skill, whether it is met, weak or missing and by how many levels.
- Compute a readiness score and a count of skills to improve.
- Generate an ordered, level-by-level improvement roadmap and a downloadable skill-analysis report.
- Keep the system modular so that new roles and skills can be added by extending the database, without changing the code.

2 Related Work and Problem Identification

Existing support for internship preparation falls into three groups. **Job portals** publish internship roles and list their requirements, but the lists are free text and differ between employers. **Learning platforms** offer courses and tutorials, but they do not tell a student which course matters for a chosen role. **Skill-assessment tools** estimate what a learner can already do, but they are not tied to the requirements of a particular internship. Each group solves part of the problem; none returns a role-specific comparison between what is required and what the student has.

2.1 Problem Statement

Students may not clearly understand the skills expected for a particular internship role. Requirements are scattered across platforms, so missing or weak skills are hard to identify. Manual comparison is time-consuming and confusing. A structured system is needed that compares a student's current skills with role requirements and presents the gaps clearly.

2.2 Gap Identification

The gap addressed by InternReady is the missing link between the three groups above: a single workflow that (i) fixes the requirements of a role in a machine-readable form, (ii) captures the student's level in each skill, (iii) quantifies the difference, and (iv) turns the difference into an ordered plan. Because the comparison uses levels and not a yes/no tick, it can also report *partial* knowledge: a student who knows HTML/CSS at Beginner level when Advanced is required has a gap of two levels, not a missing skill.

3 Proposed System Architecture

InternReady follows a layered web-application architecture (Fig. 1). The **presentation layer** is built with HTML and CSS and contains the student-profile form, the skill-rating controls and the report screens. The **application layer** is written in Python; it validates input, controls the workflow and calls the analysis engine. The **analysis engine** implements the matching, classification, scoring and roadmap rules of Section 4. The **data layer** is a MySQL database that stores students, internship roles, skills, the required level of each skill in each role, and generated reports.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

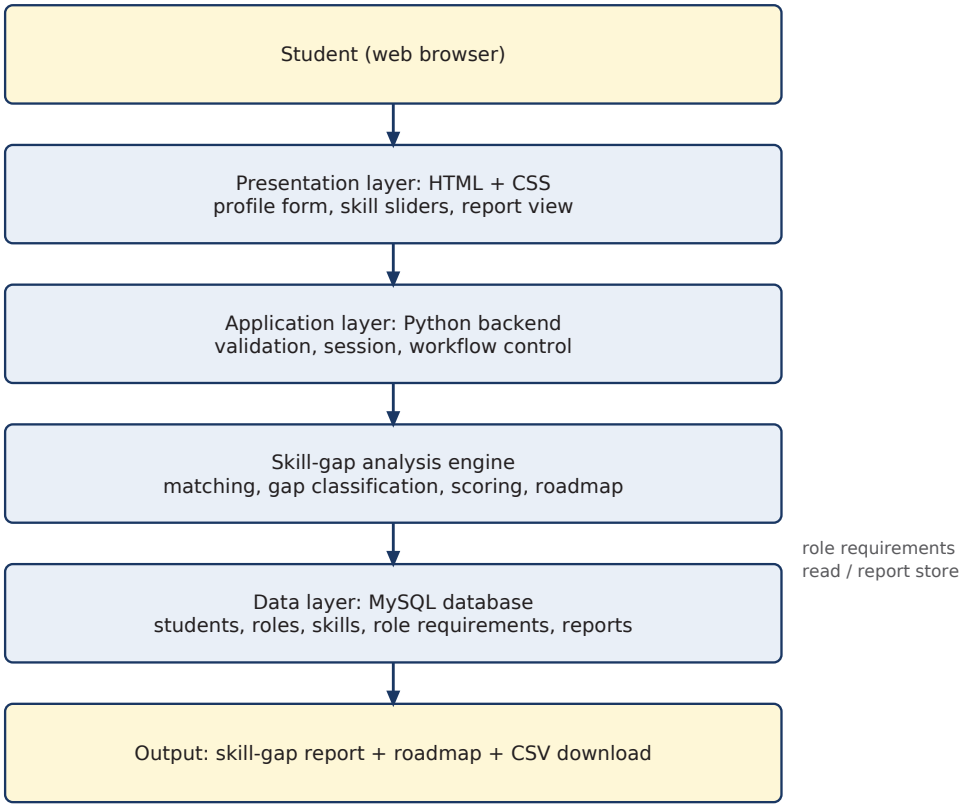


Fig. 1. Layered architecture of InternReady (data flows from the student, through the layers, to the report).

3.1 Functional Requirements

Table 1. Functional requirements of InternReady.

ID	Requirement
FR1	Create a student profile with name, branch and year of study.
FR2	Select a target internship role from the list of configured roles.
FR3	Display the required skills of the selected role with the required level of each.
FR4	Let the student rate each required skill on the 0-3 scale (Not Learned, Beginner, Intermediate, Advanced).
FR5	Compare student levels with required levels and classify each skill as meeting, weak or missing.
FR6	Compute a readiness score and the number of skills to improve.
FR7	Show a skill-comparison chart (required level against the student's level).
FR8	Generate a personalised learning roadmap ordered by size of gap.
FR9	List the student's strengths (skills at or above the required level).
FR10	Download the skill-analysis report as a file.

3.2 Non-Functional Requirements

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

Table 2. Non-functional requirements.

Attribute	Requirement
Usability	Single-page guided flow of three steps (profile, rate skills, report); short level guide shown beside the rating controls.
Extensibility	Roles, skills and required levels are rows in the database; adding a role needs no code change.
Performance	The comparison involves only a handful of skills per role, so a report is expected to be produced in well under a second on an ordinary server (to be measured; see Section 6).
Portability	Runs in a standard desktop or mobile web browser; no installation on the student's device.
Privacy	Only name, branch, year and skill levels are collected; no passwords, contact details or documents are required.
Transparency	The report states that the score is an estimate based on self-reported skill levels and on the configured role requirements.

3.3 Technical Specifications

Table 3. Technical specification summary.

Component	Specification
Presentation layer	HTML5 and CSS3 [3], [4]; responsive layout; sliders for level input; bar chart for comparison.
Application layer	Python 3 [1]; handles profile input, role lookup, calls to the analysis engine, report export.
Analysis engine	Pure Python functions: level comparison, gap classification, score computation, roadmap ordering (Section 4).
Database	MySQL 8.0 [2]; relational schema of six tables (Section 3.4).
Skill-level scale	Integer 0 to 3: 0 Not Learned, 1 Beginner, 2 Intermediate, 3 Advanced.
Inputs	Student name (text), branch (list), year (list), target role (list), one level per required skill.
Outputs	Readiness score (%), count of skills to improve, count of skills meeting the requirement, comparison chart, strengths list, roadmap, downloadable report.
Client requirement	Any current web browser with JavaScript-capable rendering of the chart; internet connection to the server.
Server requirement	A machine able to run Python 3 and MySQL 8.0; no GPU or special hardware.

3.4 Database Design

The proposed schema has six tables (Fig. 2, Table 4). *students* holds the profile. *internship_roles* and *skills* are the two catalogues. *role_skills* is the link table that stores the required level of each skill for each role; it is what allows the same skill (for example Git) to be required at different depths in different roles. *student_skills* stores the levels a student entered, and *reports* stores each generated analysis so that results can be reviewed later.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

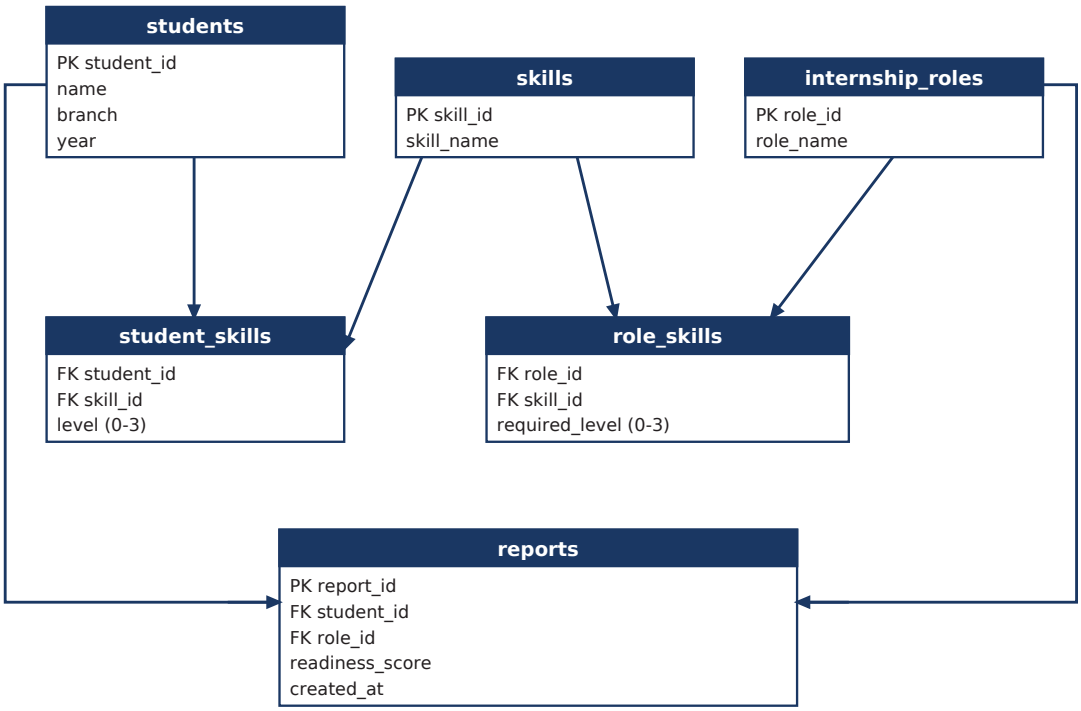


Fig. 2. Entity-relationship diagram of the proposed MySQL schema.

Table 4. Proposed database tables.

Table	Columns	Purpose
students	student_id (PK), name, branch, year	Student profile
internship_roles	role_id (PK), role_name	Catalogue of supported roles
skills	skill_id (PK), skill_name	Catalogue of skills
role_skills	role_id (FK), skill_id (FK), required_level (0-3)	Required level of each skill per role
student_skills	student_id (FK), skill_id (FK), level (0-3)	Self-rated level per skill
reports	report_id (PK), student_id (FK), role_id (FK), readiness_score, created_at	Stored analysis results

4 Methodology

The methodology follows the six steps of the project plan (profile creation, role selection, requirement analysis, skill comparison, gap identification and roadmap generation), with readiness scoring added between gap identification and roadmap generation (Fig. 3). The sub-sections below define each step precisely so that the analysis is reproducible.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

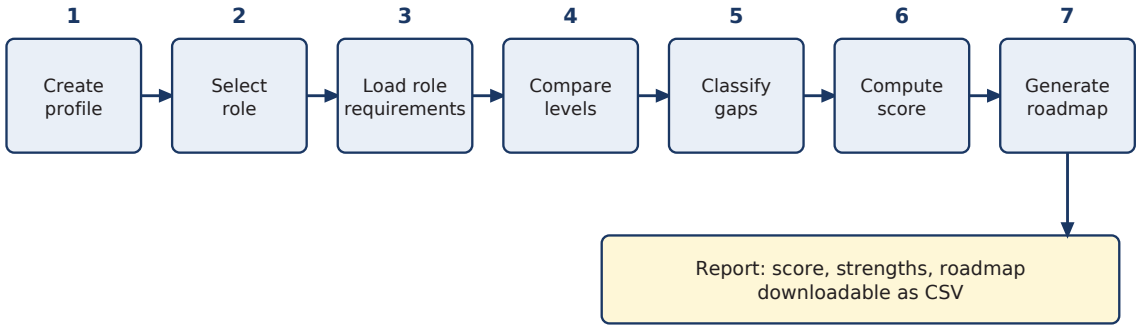


Fig. 3. Workflow of InternReady from profile creation to report.

4.1 Skill Levels and Role Requirement Model

Every skill is expressed on an ordinal scale of four levels: 0 Not Learned, 1 Beginner, 2 Intermediate and 3 Advanced. A role R is defined by a set of required skills K_R and, for each skill k in K_R, a required level r_k in {1, 2, 3}. A student's profile assigns a level s_k in {0, 1, 2, 3} to each skill in K_R. Table 5 shows the configuration used for the Full Stack Developer Intern role in the prototype and in the worked example of Section 6.2.

Table 5. Sample role configuration: Full Stack Developer Intern.

Skill	Required level r_k	Meaning of required level
HTML/CSS	3	Advanced
JavaScript	3	Advanced
React	2	Intermediate
SQL	2	Intermediate
Git	3	Advanced

4.2 Skill Matching and Gap Classification

For each skill k in K_R the gap is $g_k = \max(r_k - s_k, 0)$. The skill is then classified by three rules:

- **Meets requirement** (strength) if $s_k \geq r_k$, i.e. $g_k = 0$.
- **Weak** if $0 < s_k < r_k$: the student has started the skill but is below the required depth.
- **Missing** if $s_k = 0$: the skill has not been learned.

Weak and missing skills together form the set of *skills to improve*; their number is shown on the report. Skills that a student has but the role does not require (for example HTML and CSS for a Python role) are reported as *additional skills* and do not affect the score.

4.3 Readiness Score

The readiness score is the share of the required skill depth that the student already covers:

$$\textbf{Readiness (\%)} = 100 \times \Sigma_k \min(s_k, r_k) / \Sigma_k r_k$$

Taking the minimum prevents a skill above the requirement from hiding a gap elsewhere, so the score reaches 100% only when every requirement is met. Weights per skill could be introduced in the same formula if some skills matter more for a role; the prototype's exact weighting should be stated alongside any score that is reported. The score is an estimate because the levels are self-reported.

4.4 Roadmap Generation

Each skill to improve becomes a roadmap item of the form "*skill* — g_k level(s) to improve" (Fig. 8). Items are ordered by decreasing gap; ties keep the order of the role definition. The report

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

also lists the strengths and offers a download of the full analysis.

4.5 Algorithm

Algorithm 1. Skill-gap analysis.

```
Input : student levels s[k], role requirements r[k], k in K_R
Output: classification, readiness, roadmap

1  total_req <- 0 ; credit <- 0
2  for each skill k in K_R:
3      gap[k]   <- max(r[k] - s[k], 0)
4      credit   <- credit + min(s[k], r[k])
5      total_req<- total_req + r[k]
6      if gap[k] = 0       : class[k] <- "meets"
7      else if s[k] = 0    : class[k] <- "missing"
8      else                : class[k] <- "weak"
9  readiness <- 100 * credit / total_req
10 to_improve <- [k for k in K_R if gap[k] > 0]
11 roadmap    <- sort(to_improve, key = gap descending, stable)
12 store report(student, role, readiness) ; return all
```

The algorithm visits every skill of the role once, so its running time grows linearly with the number of required skills; the sort in step 11 adds a term proportional to n log n for n skills to improve, which is negligible for the handful of skills per role.

5 Implementation

InternReady is implemented as a web application that can be used from a desktop or phone browser. The interface is a guided three-step flow with a sidebar for the student profile. Table 6 lists the modules and Figures 4 to 8 show the working prototype.

Table 6. Modules of the InternReady prototype.

Module	Description
Student profile	Sidebar form: name, branch (for example CSE) and year (for example 2nd Year).
Role selection	Choice of target internship role, for example Full Stack Developer Intern.
Skill rating	One slider per required skill, labelled with the required level and a level guide (Not Learned = 0, Beginner = 1, Intermediate = 2, Advanced = 3).
Analysis engine	Compares levels, classifies gaps, computes score and roadmap (Section 4).
Report view	Readiness score, number of skills to improve, number of skills meeting the requirement, progress bar and skill-comparison chart.
Roadmap and strengths	Expandable roadmap items per skill; table of strengths with the student's level and the required level.
Export	Download of the skill-analysis report as a file.

5.1 User Interface Walkthrough

Step 1: student profile. The student enters a name and selects branch and year in the sidebar (Fig. 4). These values are printed on the final report.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

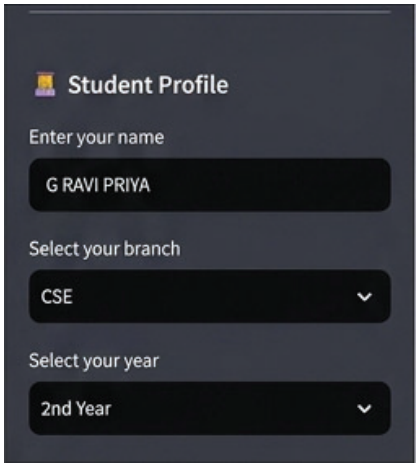


Fig. 4. Student profile panel (name, branch, year).

Step 2: rate current skills. For each skill required by the selected role the interface shows the required level (for example "Required level: 3/3") and a slider whose positions are labelled Not Learned, Beginner, Intermediate and Advanced. A level guide below the sliders explains the numeric scale (Fig. 5).

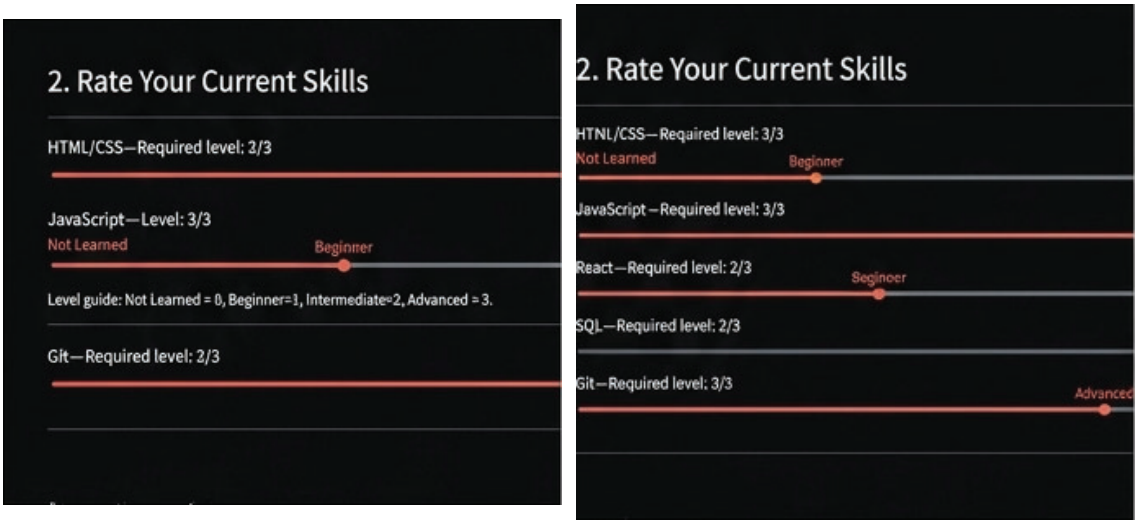


Fig. 5. Skill-rating step: sliders with the required level shown next to each skill.

Step 3: readiness report. The report header repeats the student name, branch, year and target role, and shows the readiness score, the number of skills to improve and a progress bar. A grouped bar chart places the required level and the student's level side by side for every skill (Figs. 6 and 7).

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

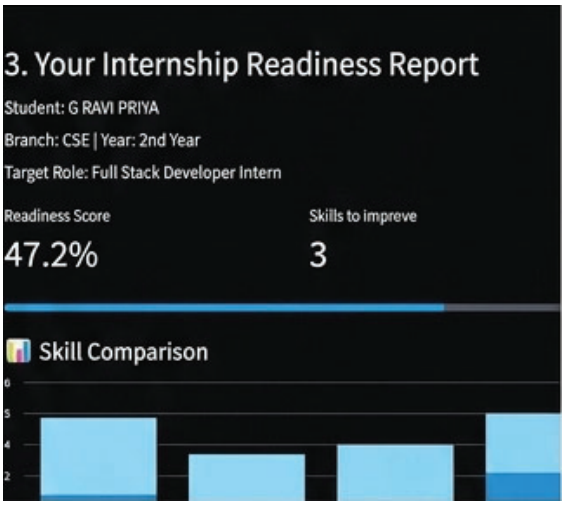
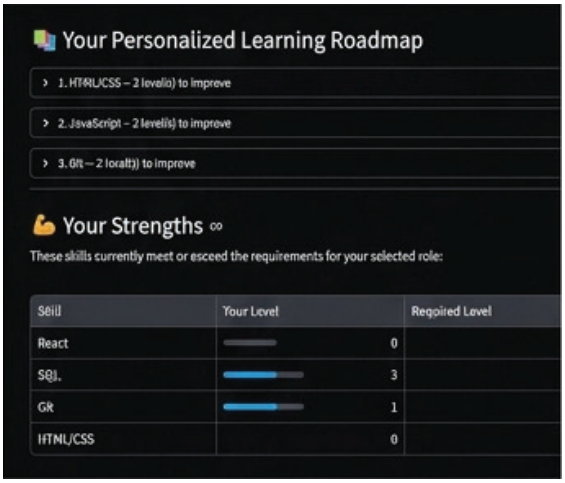




Fig. 6 (left). Report header with readiness score and skill comparison. Fig. 7 (right). Readiness score, skills to improve, skills meeting the requirement, and the comparison chart.

Roadmap, strengths and download. Below the chart the interface shows the personalised learning roadmap as expandable items (each stating how many levels to improve), the table of strengths with the student's level against the required level, and a button to download the skill-analysis report. A closing note explains that the score is an estimate based on self-reported skill levels and the configured role requirements (Fig. 8).



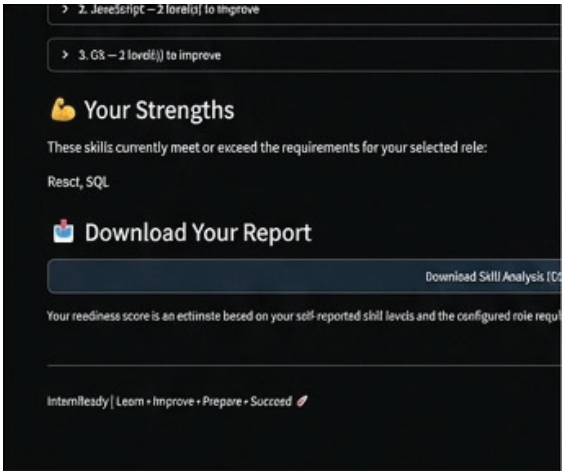


Fig. 8. Personalised learning roadmap, strengths, and report download (left: roadmap and strengths table; right: strengths list, download button and score disclaimer).

5.2 Input Validation and Privacy

Level inputs are restricted to the integers 0 to 3 by the slider control and checked again by the backend; the role and branch are chosen from fixed lists, so free-text entry is limited to the name. Only the data needed for the report is stored, and no contact details or documents are collected.

6 Testing and Results

This section distinguishes between what has been shown and what is still to be measured. The screenshots in Section 5 show that the prototype produces the full report for a sample session (readiness score 47.2%, 3 skills to improve, 2 skills meeting the requirement). The accuracy of the matching logic and the usability of the tool have not yet been measured on a test set or with users, and no such results are claimed here.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

6.1 Test Plan

Table 7. Test plan (all cases to be executed and recorded before final submission).

ID	Test	Input	Expected result	Status
T1	Exact match	All levels equal to required	All skills "meets"; readiness 100%; no roadmap items	Planned
T2	Complete absence	All levels 0	All skills "missing"; readiness 0%; roadmap lists every skill	Planned
T3	Partial skill	Level 1, required 3	Classified "weak"; gap 2; credit 1	Planned
T4	Above requirement	Level 3, required 2	Classified "meets"; credit capped at required level	Planned
T5	Multiple roles	Same student, two roles	Different gaps and scores per role	Planned
T6	Multiple students	Different profiles, same role	Independent, correct reports	Planned
T7	Roadmap order	Gaps of 2, 2 and 1	Items ordered 2, 2, 1; ties in role order	Planned
T8	Input validation	Level outside 0-3 / empty name	Rejected with message	Planned
T9	Export	Complete session	Downloaded file matches on-screen report	Planned

6.2 Worked Examples

Example A (binary skills, Python Developer Intern). Required skills: Python, SQL, Git. Student skills: Python, HTML, CSS. Matched: Python. Skill gaps: SQL, Git. Additional skills: HTML, CSS. Coverage of required skills = 1/3 = 33.3%. This is the example from the project presentation, treated as a level-free special case of the method.

Example B (levels, Full Stack Developer Intern). With the requirements of Table 5 and the student levels below, the method gives the results of Table 8.

Table 8. Worked example of gap classification (computed from Algorithm 1; sample student levels, not a measured user).

Skill	Required level	Student level	Gap	Status	Credit min(s, r)
HTML/CSS	3 (Advanced)	2 (Intermediate)	1	Weak (below required)	2
JavaScript	3 (Advanced)	1 (Beginner)	2	Weak (below required)	1
React	2 (Intermediate)	0 (Not Learned)	2	Missing (level 0)	0
SQL	2 (Intermediate)	2 (Intermediate)	0	Meets requirement	2
Git	3 (Advanced)	3 (Advanced)	0	Meets requirement	3

Total required depth is 13 and the credited depth is 8, so the readiness score is 100 × 8 / 13 = 61.5%. Three skills need improvement (JavaScript, React, HTML/CSS) and two meet the requirement (SQL, Git). The roadmap order is JavaScript (2 levels), React (2 levels), HTML/CSS (1 level). The prototype screenshots report the same counts (3 and 2) for their sample session.

6.3 Discussion

The level-based comparison gives more information than a checklist: it separates a missing skill from a weak one and quantifies how far a weak skill is from the requirement, which is what the roadmap needs. Its quality depends on three things: the accuracy and completeness of the role requirement tables, the honesty of self-rated levels, and the correctness of the comparison. The first is a content problem solved by maintaining the database; the second is a limitation of self-assessment discussed in Section 7; the third is what test cases T1 to T7 are designed to verify.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

7 Advantages and Limitations

7.1 Advantages

- One workflow replaces manual comparison across job portals, courses and assessment tools.
- Level-based comparison distinguishes strengths, weak skills and missing skills.
- Readiness score and ordered roadmap give a clear next step.
- New roles and skills are added as database rows, without code changes.
- Simple interface that runs in a browser; only a few inputs are needed.

7.2 Limitations

- Levels are self-reported and may be over- or under-estimated; the score is therefore an estimate.
- Role requirements are configured by the team and must be kept up to date with real internship postings.
- The roadmap states the gap size but does not yet recommend specific courses or resources.
- The tool measures declared skill levels only; it does not test the skills.
- Accuracy and usability results are not yet measured (Section 6).

8 Future Scope

Planned extensions are AI-based personalised recommendations, resume-based skill extraction so that levels need not be typed in, real-time internship requirements taken from live postings, progress tracking across repeated assessments, and short skill assessments that replace or verify self-rated levels. Linking each roadmap item to a curated learning resource, adding per-skill weights to the score, and supporting more roles are further extensions.

9 Conclusion

InternReady compares a student's self-rated skill levels with the required levels of a chosen internship role and returns a classification of every skill, a readiness score and an ordered improvement roadmap. This paper has specified the architecture, the database schema, the technical requirements, the matching, scoring and roadmap rules, and a test plan, and has shown the working prototype screens. The next steps are to execute the planned tests, measure matching correctness and usability with real students, and replace the sample outputs in this paper with recorded results.

10 Individual Contributions

Table 9. Contributions of team members.

Member	Contribution
G. Ravi Priya (25E51A0596)	Problem identification, literature survey and documentation.
Sandhya (25E51A0592)	System design, presentation preparation and presentation.
G. Prem Sagar (25E51A0507)	Frontend design and development.
G. Varun (25E51A05A8)	Backend development.
K. Jaya Krishna (25E51A0512)	Database design and management.

11 Work Plan

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 11 , November - 2026

Table 10. Project work plan.

Phase	Activity
1. Problem Identification	Identify internship-preparation problems.
2. Research	Study internship requirements and existing approaches.
3. System Design	Design the workflow, architecture and database.
4. Development	Develop the frontend, backend and database.
5. Skill-Gap Module	Implement skill comparison and gap identification.
6. Testing	Test different student profiles and internship roles (Table 7).
7. Final Presentation	Analyse results and prepare documentation.

Acknowledgment

The authors thank their guide, Mrs. Jahanara Begum, and the Hyderabad Institute of Technology and Management (HITAM) for academic support and guidance.

References

[1] Python Software Foundation, "Python 3 Documentation." [Online]. Available: https://docs.python.org/3/

[2] Oracle, "MySQL 8.0 Reference Manual." [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/

[3] MDN Web Docs, "HTML: HyperText Markup Language." [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML

[4] MDN Web Docs, "CSS: Cascading Style Sheets." [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/CSS

(This work is licensed under a Creative Commons Attribution 4.0 International License.)