

# SURGE '25 Web Hackathon Problem Statement

## CampusConnect – The University Talent Finder App

### Overview

Your challenge is to build an app named **CampusConnect**, an on-campus talent discovery platform where students can connect through opportunities, whether it's for part-time work, startup gigs, academic projects, or collaborations.

App includes:

- Authentication System
- Role-based dashboards
- A functional job posting and discovery system

### The Challenge

Build a Talent Finder Platform where each user can act in two modes:

1. Talent Finder – Post jobs or opportunities.
2. Talent Seeker – Browse the available jobs, applies, or expresses interest.

The goal is to simulate an intra-university job marketplace where students can collaborate on freelance work, startups, competitions, or internships.

### Core Requirements

1. Landing Page
2. Authentication
  - Users should be able to sign up, log in, and manage their profiles.
  - Every user can act as both a **Talent Finder** (posting jobs or opportunities) and a **Talent Seeker** (browsing and applying for jobs).
  - Add **email verification**, **password reset**, and **OAuth login** (Google, GitHub, etc.).
  - Include **role switching** (Talent Finder ↔ Talent Seeker) without logging out.

### 3. Talent Finder Dashboard

- Create and manage posts e.g. Academic Projects, Startup/Collaborations, Part-time Jobs, Competitions/Hackathons Teams Search.
- Add **draft saving** for job posts.
- Option to **edit, delete, and mark post as filled**.
- Add **applicant management**: view applicants, shortlist, message them.
- Include **analytics** (number of views, applications, interest rate).

### 4. Talent Seeker Dashboard

- View all available jobs.
- Filter/search jobs by title, type, or tags.
- Add **personalized recommendations** based on skills/interests.
- Option to **save/bookmark jobs**.
- **Application status tracking** (Pending, Shortlisted, Rejected, Accepted).
- Allow **upload of resumes** or **custom proposal messages** when applying.

### 5. Database Integration

- Must use a database to store users, job posts, and applications.

### 6. Engineering Logic

- Each team must include at least one feature that demonstrates applied problem-solving or algorithmic thinking. Example ideas include: job recommendation ranking, AI-assisted matching, generate profile from resumes, profile score etc.

### 7. Chat or Messaging System

- Real-time chat between seekers and finders (use WebSockets or Firebase).
- Users should be able to enable Push notifications.

### 8. Match Score

- Show a “Match Score” of applicant’s profile with job postings. (e.g., “You match 85% of this opportunity”).

### Judging Criteria

Category	Weight	Description
Functionality	30 Points	How complete and usable the MVP is. Are core req. completed. How the

		respective app is different from others.
Design & UX	20 Points	Overall UI aesthetics, layout, and ease of use.
Scalability & Architecture	20 Points	Code structure, database setup, and potential for future growth.
Engineering Logic	15 Points	Creative integration of AI or unique features beyond the basics.
Presentation & Demo	15 Points	Clarity of explanation, pitch, and how well the idea is showcased.

### **Deliverables**

1. GitHub Repository with all code and commits.
2. README file describing app and setup.
3. Optional: Deployed link.

### **Bonus Points**

### **Evaluation Day Flow**

Setup: Teams present with project ready.

Demo: Live walkthrough of features.

Q&A: Judges ask about functionality, tech stack, or system flow.

Scoring: Based on criteria.