PROJECT ABSTRACT

Hackathon Management Portal with Judging and Leaderboard

A portal that runs a hackathon end to end: team registration, submissions, rubric-based judging and a live leaderboard. Delivered built-to-order with the working application, report, PPT and viva kit.

01. Title

Hackathon Management Portal with Judging and Leaderboard

02. Introduction / Background

College hackathons are run across a form tool, a spreadsheet of scores and a chat group: judges score in isolation, nobody trusts the final ranking, and results take hours to compile. The fix is a single system where submissions, rubrics and scores live together — judges score against weighted criteria, the leaderboard recomputes on every submission, and every score is attributable and auditable. This project builds that portal, from the public event site to the winner certificates.

03. Problem Statement

Fair, transparent judging is the operational heart of a hackathon and the part most student systems skip. Spreadsheet scoring hides bias, loses history and cannot update live. The project addresses this with a scoring engine (weighted criteria, per-judge attribution, outlier flagging), a WebSocket-pushed live leaderboard, and an audit view — making fairness visible rather than claiming to automate it.

04. Objectives

- Publish an event site: tracks, prize pool, schedule, mentors, judges and a live countdown.
- Handle team registration with invite codes, team-size caps and per-track registration.
- Collect project submissions (repo, demo link, write-up) with automatic deadline locking.
- Score via weighted rubrics (innovation, execution, impact, demo quality) with per-criterion sliders and comments.
- Push a live leaderboard over WebSocket on every submitted score, with track filters.
- Deliver the complete student kit: application source, deployment setup, report, PPT and viva Q&A.;

05. Proposed Methodology

The system is developed in three stages. (1) Event model: organizers configure tracks, rubric weights, schedule and registration windows. (2) Scoring engine: each judge's 1–10 criterion scores produce a weighted judge score; the team's score is the mean across its judges, computed transactionally; outliers (a judge far from the panel mean) are flagged.

(3) Live layer: score submission triggers a recompute and a WebSocket broadcast; results lock, winners publish per track, and certificates generate.

06. System Architecture / Working

Express + PostgreSQL store events, teams, submissions and scores; role-based JWT auth separates organizer, judge and team views. Judges get assignment queues in the scoring console; submitting recomputes the team's weighted average inside a transaction and broadcasts the new ranking (design target: visible in under 1 s). The audit view shows every score per judge per team with versioned edits; announcements broadcast to team dashboards with read receipts.

07. Hardware / Software Requirements

- Node.js 18+ with Express
- PostgreSQL (events, teams, submissions, scores)
- WebSocket support (live leaderboard push)
- Nodemailer/SMTP (invites, announcements)
- PDF generation for certificates
- Docker + docker-compose

08. Expected Outcomes

An event supporting ~500 teams and ~30 judges (expected); leaderboard updates visible in about a second after each score (design target); a documented scoring formula with a worked example; and an audit view that makes judging transparent — the exact fairness story examiners ask about.

09. Applications

- College and inter-college hackathons
- Department coding competitions and ideathons
- Corporate innovation challenges
- Online judging for project expos and demo days

10. Future Scope

- Multi-event tenancy for a year-round hackathon platform
- Video-demo hosting instead of external links
- Public voting / people's-choice award track
- Plagiarism checks across submitted repositories