PROJECT ABSTRACT

Internship Application Tracker

A personal CRM for internship hunting: kanban pipeline, deadlines, interview prep checklists and season analytics. Delivered built-to-order with the full application, report, PPT and viva kit.

01. Title

02. Introduction / Background

A typical placement season means 30–50 applications scattered across spreadsheets, email threads and job portals — and students miss online-assessment deadlines or walk into interviews without their notes. This project builds a dedicated internship application tracker: a kanban board from Applied to Offer, per-application timelines, deadline reminders, interview-prep checklists and analytics (response rate, funnel conversion, best-performing sources) that turn a chaotic hunt into a managed pipeline.

03. Problem Statement

Spreadsheets cannot remind you that an assessment closes tonight, and they cannot show that your referral applications convert twice as well as cold ones. The project addresses this with a personal CRM for internships: every application is a card with stage, deadlines, contacts and documents; the board surfaces what needs action today; and analytics reveal which strategies are actually working so effort goes where it converts.

04. Objectives

- Provide a kanban pipeline (Applied → Screening → Interview → Offer/Rejected) with drag-and-drop stages.
- Track per-application timelines, deadlines, contacts, stipends and documents.
- Send reminders for assessments, interviews and follow-ups.
- Compute season analytics: response rate, funnel conversion, source performance.
- Maintain interview-prep checklists and notes per application.
- Deliver the complete student kit: application, report, PPT and viva Q&A.;

05. Proposed Methodology

The system is developed in three stages. (1) Data model: applications, stages, timeline events, tasks and documents modeled relationally so every fact has one home. (2) Workflow UI: a kanban board as the primary surface, with detail pages holding the stage tracker, timeline and checklists; deadline reminders computed from stored dates. (3) Analytics: aggregate queries over the student's own data produce funnel, source and timing charts — descriptive statistics of the user's real pipeline, not predictions.

06. System Architecture / Working

A single-user-focused web application: a JavaScript frontend rendering the board, detail views and charts; a backend API for CRUD on applications, events and tasks plus reminder scheduling; and a relational database. Authentication is per-student; an optional export produces a CSV of the full season for the placement cell. All analytics are computed client-side or via simple aggregate queries — no ML models are involved.

07. Hardware / Software Requirements

- Server: shared hosting or small VPS
- Backend: Python (Django/Flask) or Node.js with SQLite/PostgreSQL
- Frontend: HTML5, CSS3, JavaScript with a charting library
- Email service for deadline reminders
- Modern browser; responsive layout for phone access

08. Expected Outcomes

A working tracker where a student can log applications, move them through stages, and see analytics update live. Expected behavior: overdue deadlines are highlighted, stage changes append to the timeline, and the analytics page reflects the real pipeline. Usability expectations are design targets validated by the buyer through the included user walkthrough script.

09. Applications

- Internship and placement-season tracking for students
- Job-hunt pipeline management for fresh graduates
- Freelance proposal tracking for student freelancers
- Placement cells monitoring cohort-level progress (aggregate view)

10. Future Scope

- Browser extension to capture applications from job portals in one click
- Resume-version tracking per application
- Mock-interview scheduling with peers
- Cohort analytics for placement officers (anonymized)