COLLEGE CODE : 8203

COLLEGE : AVC COLLEGE OF ENGINEERING

DEPARTMENT : INFORMATION TECHNOLOGY

STUDENT NM-ID : 8EFD03039C61BA1B601456D562D0067A

ROLL NO : 23IT106

DATE :08/09/2025

Completed the project named as Phase 1

NAME: Job Application Tracker

SUBMITTED BY,

NAME: SRINATH R

MOBILE NO:9342680867

# Phase 1 — Problem Understanding & Requirements

## Problem Statement

In the modern job market, candidates often apply to numerous companies, making it difficult to keep track of their applications, interviews, and progress. The core problems are:

- **Scattered Information:** Candidates typically track applications using spreadsheets, emails, or notes, leading to fragmented and hard-to-access data.
- **Lack of Centralized Status Tracking:** Without a dedicated system, it's hard to get an at-a-glance view of applications that are **'Applied,' 'Interviewing,'** or **'Offered.'**
- **Inefficient Review:** Manually sorting and filtering applications to follow up or prepare for interviews is time-consuming and prone to errors.

There is a need for a **centralized Job Application Tracker** that:

- Allows users to submit, view, update, and delete application details via an API (using **Node.js, Express**).
- Stores persistent and structured data (**company, status, date applied, notes**) in a database (**MongoDB**).
- Provides robust filtering by application status for efficient review.
- Includes a basic **authentication system** to ensure data privacy and separation between users.

This solution aims to deliver a **simple, secure, and user-friendly tracker** to help job seekers manage their pipeline efficiently.

## Users & Stakeholders

### *Users*

These are the people who will directly interact with the Job Application Tracker:

- **Active Job Seekers:** Individuals currently applying to multiple companies and needing an organized way to manage their pipeline.

- **Career Changers/Students:** People in a transition phase who want to log and track their early applications and interview processes.
- **Freelancers/Contractors:** Professionals tracking submissions for projects or contract roles.

### *Stakeholders*

These are individuals or groups who have an *interest* in the system's success but may not directly use it:

- **Project Development Team:** Responsible for designing, coding, testing, and maintaining the tracker (you and your team).
- **System Administrators:** Ensure the smooth running of the server, MongoDB database, and authentication.
- **Educational Institution / Faculty Guide:** Oversee progress and evaluate the project deliverables (if academic).

## User Stories

User stories capture the required functionality from the user's perspective:

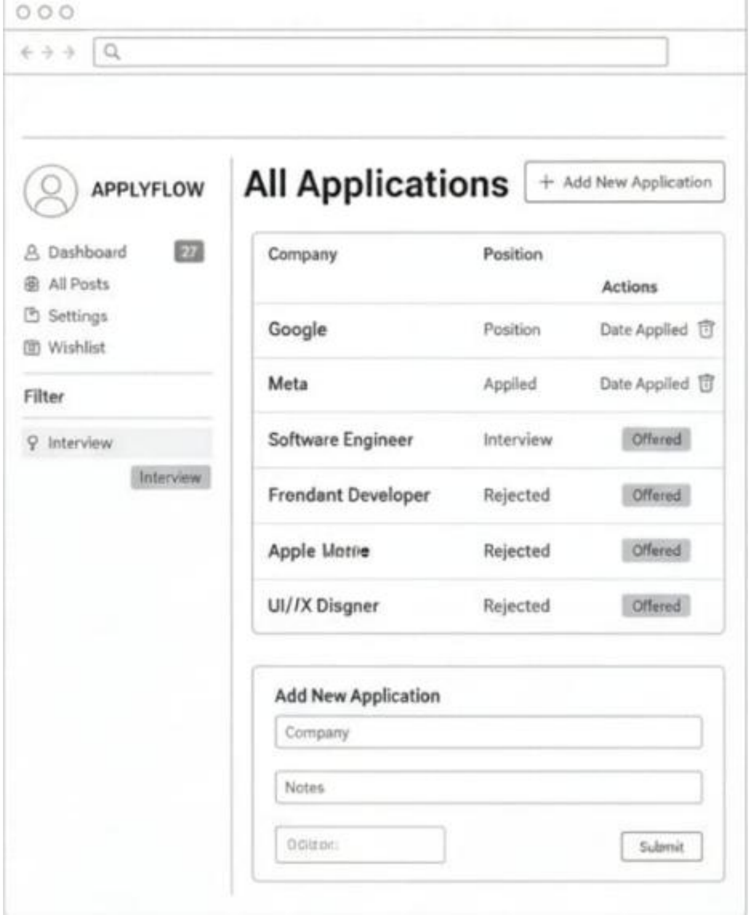
- **As a job seeker**, I want to **submit the details of a new job application** (company, date applied, notes), **so that** I can start tracking it.
- **As a job seeker**, I want to **see a list of all my applications**, **so that** I can review my overall progress.
- **As a job seeker**, I want to be able to **change the status of an application** (e.g., from 'Applied' to 'Interview'), **so that** my tracker is always up-to-date.
- **As a job seeker**, I want to be able to **filter my applications by status**, **so that** I can quickly focus on pending interviews or follow-ups.
- **As a job seeker**, I want to be able to **view and update specific notes** on an application, **so that** I can prepare for an interview or remember key follow-up points.
- **As a returning user**, I want a **secure login/authentication system**, **so that** my private application data is separate from other users' data.

## MVP [Minimum Viable Product] Features

The core features required for a functional and useful first version of the tracker:

1. **Application Submission (Create):**
    a. User can input and save **Company Name, Status, Date Applied,** and **Notes.**
2. **Application Listing & Viewing (Read):**
    a. Display a list of all current user's job applications.
    b. Allow viewing full details for a single application.
3. **Application Modification (Update):**
    a. User can update any field (e.g., change the **Status** to 'Offered').
4. **Application Deletion (Delete):**
    a. User can permanently remove an application entry.
5. **Status Filtering:**
    a. Implement filtering logic to display applications based on key statuses (e.g., 'Applied,' 'Interview,' 'Offered,' 'Rejected').
6. **Basic Authentication:**
    a. A simple login system to ensure that users can only access their own application data.

# Wireframes / API Endpoint List



All Applications

| Company | Position | Actions |
|---|---|---|
| Google | Position | Date Applied |
| Meta | Applied | Date Applied |
| Software Engineer | Interview | Offered |
| Frendant Developer | Rejected | Offered |
| Apple Motne | Rejected | Offered |
| UI/ /X Disgner | Rejected | Offered |

Add New Application

Company

Notes

## *API Endpoint List (Node.js/Express REST API)*

| Endpoint | Method | Description | Request Parameters (Body/Query) | Response |
|---|---|---|---|---|
| /api/auth/register | POST | Create a new user account. | username, password | JSON: { token } |
| /api/auth/login | POST | Authenticate and log in a user. | username, password | JSON: { token } |
| /api/applications | POST | Submit a new job application entry. | company, status, dateApplied, notes | JSON: { _id, company, ... } |
| /api/applications | GET | Retrieve a list of all user applications. | status (query, optional, for filtering) | JSON array: [ {company, status, ...}, ... ] |

| | | | | |
|---|---|---|---|---|
| /api/applications/:id | PUT | Update an existing application. | company, status, notes (as needed) | JSON: { message: "Updated successfully" } |
| /api/applications/:id | DELETE | Delete an application entry. | (None) | JSON: { message: "Deleted successfully" } |

### *Conceptual Wireframes (Description)*

For Phase 1, the wireframes should show the key interactions:

1. **Login/Register Screen:** Simple form with fields for Username and Password and buttons for Login and Register.
2. **Application List Dashboard:**
    a. A main view showing all tracked applications in a table/card format.
    b. A **Filter** section (e.g., dropdown/buttons) to select status (Applied, Interview, Offered).
    c. A **"Add New Application"** button.
    d. Each entry shows **Company Name**, **Status**, and **Date Applied**, with **"Edit"** and **"Delete"** icons.
3. **Add/Edit Application Form:**
    a. A modal or separate page with input fields for **Company Name, Date Applied, Status** (as a dropdown), and a large text area for **Notes**.
    b. **"Save"** and **"Cancel"** buttons.

## Acceptance Criteria

These criteria will determine if a feature is considered complete and functional:

### *Core CRUD Functionality*

- **Submission (POST):** A new application entry must be successfully created in MongoDB with all required fields (Company, Status, Date, Notes).
- **Retrieval (GET):** The system must successfully return *only* the applications belonging to the authenticated user.

- **Update (PUT):** Changing the status or notes of an application must be reflected correctly in the database.
- **Deletion (DELETE):** Deleting an entry must permanently remove it from the database.

### *Status Filtering*

- Using the status query parameter (e.g., /api/applications?status=Interview) must filter the list to show only applications matching that status.

### *Authentication*

- After successful login, the user must receive a token for authenticated requests.
- Unauthorized attempts to access /api/applications (without a valid token) must receive an "Access Denied" or 401 Unauthorized error.

### *Error Handling*

- **Missing Fields:** Submitting an application without the required fields (e.g., Company Name) must return a clear error message.
- **Invalid ID:** Attempting to update or delete a non-existent application ID must return a 404 "Application not found" message.
- **Data Integrity:** The application status must be one of the defined values (e.g., Applied, Interview, Offered, Rejected) and reject invalid input.