COLLEGE CODE : 2603

COLLEGE NAME : GOVERNMENT COLLEGE OF ENGINEERING BARGUR.

DEPARTMENT : COMPUTER SCIENCE AND ENGINEERING.

STUDENT NM ID : be3232c847051d090ea417a3e6e495ad

ROLL NUMBER : 2303610710422002.

DATE : 07.10.2025.

Completed the project named as phase 5. .

TECHNOLOGY PROJECT NAME : JOB APPLICATION TRACKER

SUBMITTED BY,

NAME:AFRIN FATHIMA N.
MOBILE NUMBER:8925698271.

Job Application Tracker

1. Final Demo Walkthrough

A complete demonstration of the Job Application Tracker system was conducted to highlight its working features and usability.

The demo begins with a brief introduction to the purpose of the project – helping job seekers efficiently track, organize, and manage their job applications.

The system flow is demonstrated step-by-step:

1. User Login/Signup: Users can create an account or log in securely.

2. Dashboard: Displays a summary of all job applications (applied, shortlisted, interviewed, rejected, etc.).

3. Add Application: Allows users to enter job details such as company name, role, application date, and current status.

4. Edit / Delete Options: Users can update or remove any job entry easily.

5. Filter and Search: Enables quick search by company, job title, or status.

6. Statistics & Reports: Shows analytics on total applications, success rate, and job stages.

2. Project Report Preparation

The project report was prepared to document every aspect of development, from planning to deployment.

It includes the following sections:

Abstract: Overview of the Job Application Tracker and its purpose.

Problem Statement: Difficulty faced by job seekers in tracking multiple job applications manually.

Proposed Solution: A web-based tracker to manage applications efficiently.

System Design: Architecture diagram, data flow diagram, and database schema showing how data is stored and managed.

Implementation: Technologies used – HTML, CSS, JavaScript, React.js (Frontend), Node.js / Express (Backend), MongoDB (Database).

Testing: Test cases for adding, updating, filtering, and deleting applications.

Deployment: Application hosted using Vercel (frontend) and MongoDB Atlas for the backend database.

3. Screenshots / API Documentation

Source code:

```
from flask import Flask, render_template, request, redirect, url_for

from models import db, JobApplication

from datetime import datetime


app = Flask(__name__)

app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///applications.db'

app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

db.init_app(app)


# Home/Dashboard
```

```
@app.route('/')
def dashboard():
    applications = JobApplication.query.all()
    return render_template('dashboard.html', applications=applications)


# Add Job Application
@app.route('/add', methods=['GET', 'POST'])
def add_application():
    if request.method == 'POST':
        new_app = JobApplication(
            company=request.form['company'],
            role=request.form['role'],
            platform=request.form['platform'],
            date_applied=datetime.strptime(request.form['date_applied'], "%Y-%m-
%d"),
            deadline=datetime.strptime(request.form['deadline'], "%Y-%m-%d"),
            stage=request.form['stage'],
            notes=request.form['notes']
        )
        db.session.add(new_app)
        db.session.commit()
        return redirect(url_for('dashboard'))
    return render_template('add.html')
```

```
# Update Job Application
@app.route('/update/<int:id>', methods=['GET', 'POST'])
def update_application(id):
    app_data = JobApplication.query.get(id)
    if request.method == 'POST':
        app_data.company = request.form['company']
        app_data.role = request.form['role']
        app_data.platform = request.form['platform']
        app_data.date_applied = datetime.strptime(request.form['date_applied'], "%Y-%m-%d")
        app_data.deadline = datetime.strptime(request.form['deadline'], "%Y-%m-%d")
        app_data.stage = request.form['stage']
        app_data.notes = request.form['notes']
        db.session.commit()
        return redirect(url_for('dashboard'))
    return render_template('add.html', app=app_data)

# Delete Job Application
@app.route('/delete/<int:id>')
def delete_application(id):
    app_data = JobApplication.query.get(id)
```

```
    db.session.delete(app_data)
    db.session.commit()
    return redirect(url_for('dashboard'))

if __name__ == "__main__":
    with app.app_context():
        db.create_all()
    app.run(debug=True)
```

Job Application Tracker

Company	Role	Stage	
Google	Software Eng.	Interview	Edit Dele
Amazon	Data Analyst	Applied	Edit Dele
Infosys	Intern	Selected	Edit Dele

Add New Application

Company

Role

Platform

Date Applied

YYYY-MM-DD

Deadline

YYYY-MM-D

Stage

Notes

Save Application

4. Challenges & Solutions

1. Database Integration – Faced issues connecting Flask with SQLite.

   Solved by using SQLAlchemy ORM for smooth CRUD operations.

2. Form Validation – Invalid inputs caused errors.

   Used HTML required fields and backend checks.

3. Update/Delete Routes – Trouble handling record IDs.

   Used dynamic routes like /update/<id> and /delete/<id>.

4. Template Display – Data not updating after changes.

   Used Jinja2 templates and page redirection after commit.

5. UI/UX Design – Pages looked plain.

   Added Bootstrap for a clean, user-friendly layout.

6. Date Formatting – Input and database format mismatch.

   Used datetime.strptime() for consistency.

7. Deployment – Hosting Flask on free platforms was tricky.

   Used Render/PythonAnywhere for cloud deployment.

5. GitHub README & Setup Guide

A detailed README.md file was created in the GitHub repository with the following details:

Project Title: Job Application Tracker

Description: A web app to manage and monitor job applications effectively.

Tech Stack: React.js, Node.js, Express, MongoDB, and Vercel.

Features: Add, edit, delete, and track job applications; filter by status; view analytics.

Installation Steps:

1. Clone repository: git clone <https://github.com/Afrinhussain/Job-application-tracker>

2. Install dependencies: npm install

3. Run backend server: npm start

4. Run frontend: npm run dev or npm start

6. Final Submission (Repository + Deployed Link)

At the end of Phase 5, all deliverables were submitted for evaluation:

GitHub Repository: Contains full source code and documentation.

README File: Includes project details, setup guide, and live link.

Deployed Link: https://github.com/Afrinhussain/Job-application-tracker

Final Project Report: Submitted in PDF format.

Demo Presentation: Conducted to demonstrate all project functionalities.