November 24, 2024

Final Report:

Job Recruitment Database System

GROUP7

Course: Databases CSCI 4710
Prof.: Mainul Mamun, Ph.D.

BY
Vigneshwar Reddy Gangireddy – vigneshwarreddy.gangireddy@slu.edu
Alex Chilaka – alex.chilaka@slu.edu
Sumanth Kothamasu – Sumanth.kothamasu@slu.edu
Bharath Bandlamudi – bandlamudi.bharath@slu.edu
Kasi Viswanadh Mogali – kasiviswanadh.mogali@slu.edu

Final Report

Project Title: Job Recruitment Database System

1. Problem Definition & Requirements

Problem Statement:

The objective of this project is to design, implement, and integrate a relational database system to handle job postings, applicant information, and applications. In the context of a recruitment platform, the system should address the immediate need for an efficient and scalable solution that is supportive of key operations around posting jobs, applying for jobs, updating user profiles, and tracking application statuses.
This solution solves the problem of not having a centralized, automated database system to efficiently store and manage job openings, applicant profiles, and application data. A manual or ad hoc system would be prone to errors, redundancies, and inefficiencies, leading to problems in tracking the status of applications, maintaining consistency of data, and ensuring a user-friendly experience.

Requirements Gathering:

The functional requirements for the project include:

- Ability for admins to post, update, and delete job postings.
- Applicants can create profiles, apply for jobs, and update their profiles.
- Admins and applicants can view job listings and application statuses.
- The system must handle skills and applicants' skill sets, ensuring easy modification and retrieval.

Scope and Feasibility:

The scope of the project includes:

- Design of an ERD that covers all major entities like Admin, Applicant, Job Posting, Application, Skills, etc.
- A fully functional SQL database that can handle CRUD operations efficiently.
- A basic user interface for interacting with the database.

The project is feasible given the time constraints and available resources. The database design is scalable and can be extended with additional features, such as recommendation engine for applicants.

2. Database Design

Entity Relationship Diagram (ERD):

The ERD includes the following major entities:

- **Admin:** represents companies posting jobs. Includes attributes like company name, location, and industry.
- **Applicant**: Represents users applying for jobs, with attributes such as name, contact information, resume links, and experience.
- **Job posting:** represents a job opportunity with attributes such as job title, description, salary, location, and required skills.
- **Application**: represents the application made by an applicant for a particular job.
- **Skills**: Represents individual skills available in the system, such as Python, SQL, Machine Learning, etc.

Relationships:

Relationships include:

- Admin posts job postings.
- Applicants can apply to job postings.
- Applicants can have multiple skills.

Normalization:

The database for the Job Application System has been normalized up to the Third Normal Form (3NF) to ensure data integrity and avoid redundancy. The normalization process includes the following steps, tailored to the project:

1. **First Normal Form (1NF):**
    - Ensured that all attributes in each table contain only atomic values.
    - Example: Skills are stored as individual records in the Skills table rather than as a comma-separated list in the Applicant table.

2. **Second Normal Form (2NF):**
    - Removed partial dependencies by creating separate tables for attributes dependent on only part of a composite key.
    - Example: Applicant skills were moved to a separate Applicant_Skills table to ensure proper representation of the many-many relationship between the applicant and skills.

3. **Third Normal Form (3NF):**

    - Eliminated transitive dependencies by ensuring that non-key attributes are dependent only on the primary key.

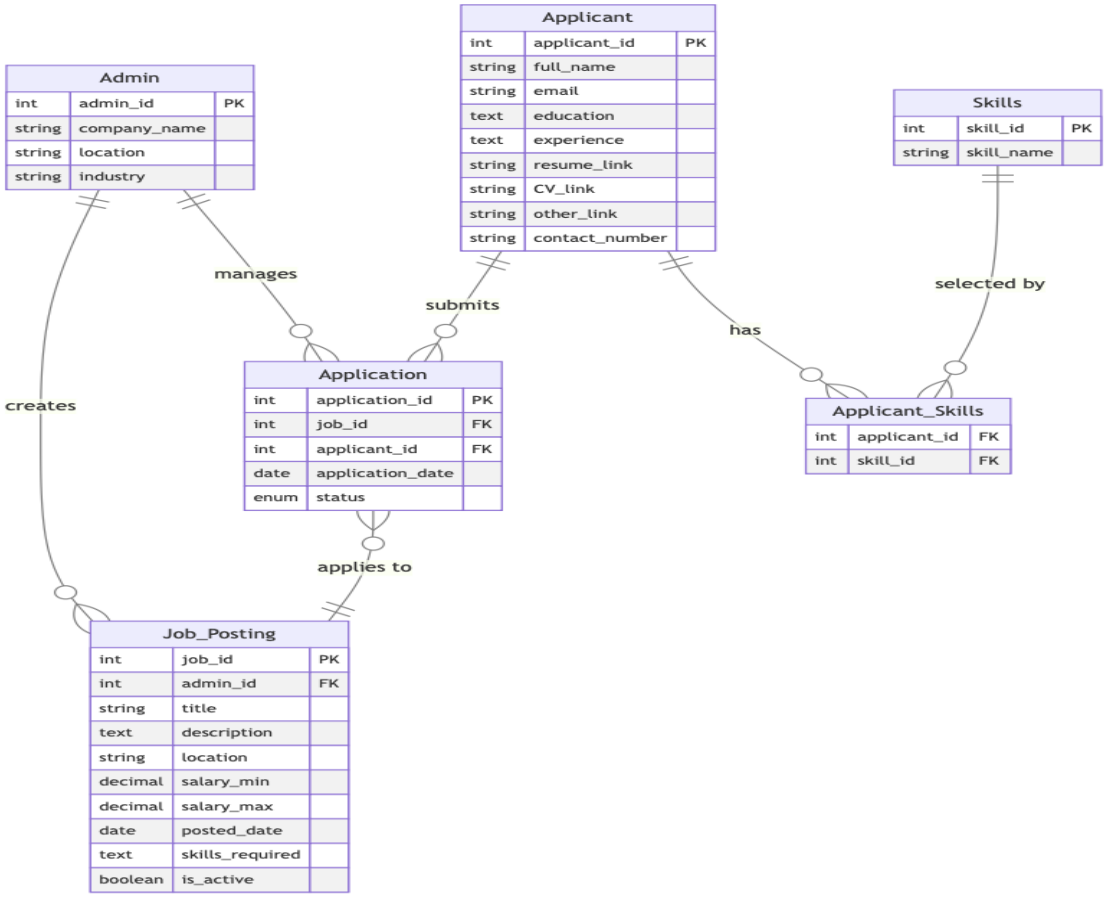
- Example: The Job Posting table includes attributes like job_id, job_title, and job_description directly linked to the job_id primary key, without indirect dependencies.

By normalizing the database to 3NF, the system achieves:

- **Minimizing Redundancy:** Each piece of information is stored only once, preventing data duplication.
- **Data Integrity:** Referential integrity is maintained across the tables using primary and foreign keys.
- **Efficient Queries:** The optimized structure allows for the fastest data retrieval and updates.

This normalization ensures the database is well-structured to handle the complexities of the job application system effectively.

ER Diagram:



3. Implementation

SQL queries:

The following sql queries were implemented:

- **Table creation scripts**

drop database job_recruitment;

```
CREATE DATABASE job_recruitment;

USE job_recruitment;
-- User Table
CREATE TABLE Admin (
    admin_id INT AUTO_INCREMENT PRIMARY KEY,
    company_name VARCHAR(100),
    location VARCHAR(100),
    industry VARCHAR(50)
);

CREATE TABLE Applicant (
    applicant_id INT AUTO_INCREMENT PRIMARY KEY,
    full_name VARCHAR(100),
    email VARCHAR(100),
    education VARCHAR(200),
    experience TEXT,
    resume_link VARCHAR(255),
    CV_link VARCHAR(255),
    other_link VARCHAR(255),
    contact_number VARCHAR(20)
);


CREATE TABLE Job_Posting (
    job_id INT AUTO_INCREMENT PRIMARY KEY,
    admin_id INT,
    title VARCHAR(100),
    description TEXT,
    location VARCHAR(100),
    salary_min DECIMAL(10, 2),
    salary_max DECIMAL(10, 2),
    posted_date DATE,
    skills_required TEXT,
    is_active BOOLEAN DEFAULT TRUE,
    FOREIGN KEY (admin_id) REFERENCES Admin(admin_id) ON DELETE CASCADE
);

CREATE TABLE Application (
    application_id INT AUTO_INCREMENT PRIMARY KEY,
    job_id INT,
    applicant_id INT,
    application_date DATE,
    status ENUM('Submitted', 'Reviewed', 'Interviewed', 'Offered', 'Rejected'),
    FOREIGN KEY (job_id) REFERENCES Job_Posting(job_id) ON DELETE CASCADE,
    FOREIGN KEY (applicant_id) REFERENCES Applicant(applicant_id) ON DELETE CASCADE
);

CREATE TABLE Skills (
    skill_id INT AUTO_INCREMENT PRIMARY KEY,
    skill_name VARCHAR(50)
```

```
);

CREATE TABLE Applicant_Skills (
    applicant_id INT,
    skill_id INT,
    PRIMARY KEY (applicant_id, skill_id),
    FOREIGN KEY (applicant_id) REFERENCES Applicant(applicant_id) ON DELETE CASCADE,
    FOREIGN KEY (skill_id) REFERENCES Skills(skill_id) ON DELETE CASCADE
);
```

✓	1	18:28:13	CREATE DATABASE job_recruitment	1 row(s) affected	0.0098 sec
✓	2	18:28:13	USE job_recruitment	0 row(s) affected	0.00021 sec
✓	3	18:28:13	CREATE TABLE Admin (admin_id INT AUTO_INCREMENT PRIMARY KEY, company_name VARCHAR(100), locati...	0 row(s) affected	0.014 sec
✓	4	18:28:13	CREATE TABLE Applicant (applicant_id INT AUTO_INCREMENT PRIMARY KEY, full_name VARCHAR(100), email...	0 row(s) affected	0.0020 sec
✓	5	18:28:13	CREATE TABLE Resume (resume_id INT AUTO_INCREMENT PRIMARY KEY, file_name VARCHAR(255), file_pat...	0 row(s) affected	0.0035 sec
✓	6	18:28:13	CREATE TABLE Cover_Letter (cover_letter_id INT AUTO_INCREMENT PRIMARY KEY, file_name VARCHAR(255),...	0 row(s) affected	0.0019 sec
✓	7	18:28:13	CREATE TABLE Job_Posting (job_id INT AUTO_INCREMENT PRIMARY KEY, admin_id INT, title VARCHAR(100),...	0 row(s) affected	0.0034 sec
✓	8	18:28:13	CREATE TABLE Application (application_id INT AUTO_INCREMENT PRIMARY KEY, job_id INT, applicant_id INT,...	0 row(s) affected	0.0055 sec
✓	9	18:28:13	CREATE TABLE Skills (skill_id INT AUTO_INCREMENT PRIMARY KEY, skill_name VARCHAR(50))	0 row(s) affected	0.0014 sec
✓	10	18:28:13	CREATE TABLE Applicant_Skills (applicant_id INT, skill_id INT, PRIMARY KEY (applicant_id, skill_id), FOREIG...	0 row(s) affected	0.0030 sec

Sample Queries:

```
INSERT INTO Admin (company_name, location, industry)
VALUES ('Innovate AI', 'Seattle, WA', 'Technology');

-- 2. Add Applicant Example
INSERT INTO Applicant (full_name, email, education, experience, resume_link, CV_link,
other_link, contact_number)
VALUES ('Lisa Brown', 'lisa.brown@email.com', 'MS Computer Science', '7 years ML
engineering',
        'resume.com/lbrown', 'cv.com/lbrown', 'github.com/lbrown', '678-901-2345');

-- 3. Post New Job Example
INSERT INTO Job_Posting (admin_id, title, description, location, salary_min, salary_max,
posted_date, skills_required, is_active)
VALUES (1, 'ML Engineer', 'Seeking experienced ML engineer for NLP projects', 'San
Francisco, CA',
        130000, 190000, CURDATE(), 'Python, Machine Learning, NLP', TRUE);

-- 4. View Posted Jobs (All Active Jobs)
SELECT
    j.job_id,
    j.title,
    j.description,
    j.location,
    j.salary_min,
    j.salary_max,
    j.posted_date,
    j.skills_required,
    a.company_name,
    a.industry
FROM Job_Posting j
JOIN Admin a ON j.admin_id = a.admin_id
WHERE j.is_active = TRUE
ORDER BY j.posted_date DESC;

-- 5. Search Jobs Example (searching for 'Python' in all fields)
SELECT
    j.job_id,
    j.title,
    j.description,
    j.location,
```

```
    j.salary_min,
    j.salary_max,
    j.posted_date,
    j.skills_required,
    a.company_name,
    a.industry
FROM Job_Posting j
JOIN Admin a ON j.admin_id = a.admin_id
WHERE j.is_active = TRUE
AND (
    j.title LIKE '%Python%' OR
    a.company_name LIKE '%Python%' OR
    j.location LIKE '%Python%' OR
    j.skills_required LIKE '%Python%'
)
ORDER BY j.posted_date DESC;

-- 6. Filter Jobs Example (specific location and salary range)
SELECT
    j.job_id,
    j.title,
    j.description,
    j.location,
    j.salary_min,
    j.salary_max,
    j.posted_date,
    j.skills_required,
    a.company_name,
    a.industry
FROM Job_Posting j
JOIN Admin a ON j.admin_id = a.admin_id
WHERE j.is_active = TRUE
AND (j.location = 'San Francisco, CA')
AND (a.industry = 'Technology')
AND (j.salary_min >= 100000)
AND (j.salary_max <= 200000)
ORDER BY j.posted_date DESC;

-- 7. Apply for Jobs Example
INSERT INTO Application (job_id, applicant_id, application_date, status)
VALUES (1, 3, CURDATE(), 'Reviewed');

-- 8. View Applications with Applicant Skills Example (for admin_id = 1)
SELECT
    app.application_id,
    app.application_date,
    app.status,
    j.title AS job_title,
    j.location AS job_location,
    a.company_name,
    ap.full_name AS applicant_name,
    ap.email,
    ap.education,
    ap.experience,
    ap.resume_link,
    ap.CV_link,
    GROUP_CONCAT(s.skill_name) AS skills
FROM Application app
JOIN Job_Posting j ON app.job_id = j.job_id
JOIN Admin a ON j.admin_id = a.admin_id
```

```
JOIN Applicant ap ON app.applicant_id = ap.applicant_id
LEFT JOIN Applicant_Skills aps ON ap.applicant_id = aps.applicant_id
LEFT JOIN Skills s ON aps.skill_id = s.skill_id
WHERE j.admin_id = 1
GROUP BY app.application_id;

-- 9. Update Application Status Example
UPDATE Application
SET status = 'reviewed'
WHERE application_id = 1;
-- 10. View Application Status Example (for applicant_id = 1)
SELECT
    a.application_id,
    j.title,
    ad.company_name,
    a.application_date,
    a.status
FROM Application a
JOIN Job_Posting j ON a.job_id = j.job_id
JOIN Admin ad ON j.admin_id = ad.admin_id
WHERE a.applicant_id = 1
ORDER BY a.application_date DESC;

-- 11. Manage Admins Examples
-- 11.1 View All Admins
SELECT * FROM Admin;

-- 11.2 Update Admin Example
UPDATE Admin
SET company_name = 'Tech Solutions International',
    location = 'San Francisco, CA',
    industry = 'Technology'
WHERE admin_id = 1;

-- 11.3 Delete Admin Example
DELETE FROM Admin WHERE admin_id = 5;

-- 12. Update Applicant Profile Example
UPDATE Applicant
SET full_name = 'John Smith Jr',
    email = 'john.smith.jr@email.com',
    education = 'MS Computer Science',
    experience = '6 years software development',
    resume_link = 'resume.com/jsmith_jr',
    CV_link = 'cv.com/jsmith_jr',
    other_link = 'github.com/jsmith_jr',
    contact_number = '123-456-7891'
WHERE applicant_id = 1;

-- 13. Manage Applicant Skills Examples
-- 13.1 Add new skill
INSERT INTO Applicant_Skills (applicant_id, skill_id)
VALUES (1, 8);  -- Adding Machine Learning skill to John Smith

-- 13.2 Delete skill Example
DELETE FROM Applicant_Skills
WHERE applicant_id = 1 AND skill_id = 2;  -- Remove JavaScript skill from John Smith

-- 13.3 View applicant's current skills Example
SELECT s.skill_id, s.skill_name
```

```
FROM Skills s
JOIN Applicant_Skills aps ON s.skill_id = aps.skill_id
WHERE aps.applicant_id = 1;

-- 13.4 Update all skills for an applicant Example
START TRANSACTION;
    -- Delete all current skills for John Smith
    DELETE FROM Applicant_Skills WHERE applicant_id = 1;

    -- Insert new skills for John Smith
    INSERT INTO Applicant_Skills (applicant_id, skill_id)
    VALUES
    (1, 1),  -- Python
    (1, 3),  -- SQL
    (1, 8);  -- Machine Learning
COMMIT;
```

26	17:12:34	INSERT INTO Admin (company_name, location, industry) VALUES ('Innovate AI', 'Seattle, WA', 'Technology')	1 row(s) affected	
27	17:12:36	INSERT INTO Applicant (full_name, email, education, experience, resume_link, CV_link, other_link, contact_number) VALUES ('Lisa Brown', 'lisa.brown@email.com', '...	1 row(s) affected	
28	17:12:38	INSERT INTO Job_Posting (admin_id, title, description, location, salary_min, salary_max, posted_date, skills_required, is_active) VALUES (1, 'ML Engineer', 'Seeking...	1 row(s) affected	
29	17:12:40	SELECT j.job_id, j.title, j.description, j.location, j.salary_min, j.salary_max, j.posted_date, j.skills_required, a.company_name, a.industry FRO...	5 row(s) returned	
30	17:12:43	SELECT j.job_id, j.title, j.description, j.location, j.salary_min, j.salary_max, j.posted_date, j.skills_required, a.company_name, a.industry FRO...	4 row(s) returned	
31	17:12:48	SELECT j.job_id, j.title, j.description, j.location, j.salary_min, j.salary_max, j.posted_date, j.skills_required, a.company_name, a.industry FRO...	1 row(s) returned	
32	17:12:51	INSERT INTO Application (job_id, applicant_id, application_date, status) VALUES (1, 3, CURDATE(), 'Reviewed')	1 row(s) affected	
33	17:12:56	SELECT app.application_id, app.application_date, app.status, j.title AS job_title, j.location AS job_location, a.company_name, ap.full_name AS appl...	5 row(s) returned	
34	17:12:59	UPDATE Application SET status = 'reviewed' WHERE application_id = 1	0 row(s) affected Rov	
35	17:13:03	SELECT a.application_id, j.title, ad.company_name, a.application_date, a.status FROM Application a JOIN Job_Posting j ON a.job_id = j.job_id JOIN Ad...	1 row(s) returned	
36	17:13:09	SELECT * FROM Admin LIMIT 0, 1000	5 row(s) returned	
37	17:13:12	UPDATE Admin SET company_name = 'Tech Solutions International', location = 'San Francisco, CA', industry = 'Technology' WHERE admin_id = 1	1 row(s) affected Rov	
38	17:13:14	DELETE FROM Admin WHERE admin_id = 5	1 row(s) affected	
39	17:13:16	UPDATE Applicant SET full_name = 'John Smith Jr', email = 'john.smith.jr@email.com', education = 'MS Computer Science', experience = '6 years software d...	1 row(s) affected Rov	
40	17:13:20	INSERT INTO Applicant_Skills (applicant_id, skill_id) VALUES (1, 8)	1 row(s) affected	
41	17:13:22	DELETE FROM Applicant_Skills WHERE applicant_id = 1 AND skill_id = 2	0 row(s) affected	
42	17:13:25	SELECT s.skill_id, s.skill_name FROM Skills s JOIN Applicant_Skills aps ON s.skill_id = aps.skill_id WHERE aps.applicant_id = 1 LIMIT 0, 1000	4 row(s) returned	
43	17:13:31	START TRANSACTION	0 row(s) affected	
44	17:13:34	DELETE FROM Applicant_Skills WHERE applicant_id = 1	4 row(s) affected	
45	17:13:36	INSERT INTO Applicant_Skills (applicant_id, skill_id) VALUES (1, 1), -- Python (1, 3), -- SQL (1, 8)	3 row(s) affected Rec	
46	17:13:39	COMMIT	0 row(s) affected	

Sample query Results:

job_id	title	description	location	salary_min	salary_max	posted_date	skills_required	company_name	industry
5	ML Engineer	Seeking experienced ML engineer for NLP proje…	San Francisco, CA	130000.00	190000.00	2024-11-24	Python, Machine Learning, NLP	Tech Solutions International	Technology
4	ML Engineer	Looking for an ML engineer with cloud computin…	Chicago, IL	95000.00	140000.00	2024-01-18	Python, Machine Learning, Cloud Computing	Financial Experts Ltd	Finance
3	Project Manager	Healthcare IT project manager needed for large…	Boston, MA	85000.00	120000.00	2024-01-17	Project Management, Healthcare IT	Healthcare Plus	Healthcare
2	Data Scientist	Seeking a data scientist with machine learning e…	New York, NY	90000.00	130000.00	2024-01-16	Python, Machine Learning, Data Analysis	Tech Solutions International	Technology
1	Senior Software Engineer	Looking for an experienced software engineer w…	New York, NY	100000.00	150000.00	2024-01-15	Python, JavaScript, SQL	Tech Solutions International	Technology

job_id	title	description	location	salary_min	salary_max	posted_date	skills_required	company_name	industry
5	ML Engineer	Seeking experienced ML engineer for NLP proje…	San Francisco, CA	130000.00	190000.00	2024-11-24	Python, Machine Learning, NLP	Tech Solutions International	Technology
4	ML Engineer	Looking for an ML engineer with cloud computin…	Chicago, IL	95000.00	140000.00	2024-01-18	Python, Machine Learning, Cloud Computing	Financial Experts Ltd	Finance
2	Data Scientist	Seeking a data scientist with machine learning e…	New York, NY	90000.00	130000.00	2024-01-16	Python, Machine Learning, Data Analysis	Tech Solutions International	Technology
1	Senior Software Engineer	Looking for an experienced software engineer w…	New York, NY	100000.00	150000.00	2024-01-15	Python, JavaScript, SQL	Tech Solutions International	Technology

application_...	application_da...	status	job_title	job_location	company_name	applicant_na...	email	education	experience	resume_link	CV_link	skills
1	2024-01-20	Rejected	Senior Software E...	New York, NY	Tech Solutions Inter...	John Smith Jr	john.smith.jr@email.com	MS Computer Science	6 years software development	resume.com/jsmith_jr	cv.com/jsmith_jr	Python,SQL,Cloud Computing
2	2024-01-21	Reviewed	Data Scientist	New York, NY	Tech Solutions Inter...	Jane Smith	jane.smith@email.com	Master in Data Science	3 years data analyst experience	https://storage/resume/jane_smith.pdf	https://storage/cv/jane_smith.pdf	Python,Machine Learning,Data Analysi
5	2024-01-24	Reviewed	Senior Software E...	New York, NY	Tech Solutions Inter...	Jane Smith	jane.smith@email.com	Master in Data Science	3 years data analyst experience	https://storage/resume/jane_smith.pdf	https://storage/cv/jane_smith.pdf	Python,Machine Learning,Data Analysi
6	2024-01-25	Reviewed	Data Scientist	New York, NY	Tech Solutions Inter...	Alice Brown	alice.brown@email.com	Master in Artificial Intellig...	2 years ML engineer experience	https://storage/resume/alice_brown.pdf	https://storage/cv/alice_brown.pdf	Python,Machine Learning,Cloud Comp
7	2024-11-24	Reviewed	Senior Software E...	New York, NY	Tech Solutions Inter...	Bob Wilson	bob.wilson@email.com	Bachelor in Business Ad...	4 years project management experience	https://storage/resume/bob_wilson.pdf	https://storage/cv/bob_wilson.pdf	SQL,Project Management

application_...	title	company_name	application_da...	status
1	Senior Software Engineer	Tech Solutions Inter…	2024-01-20	Reviewed

skill_id	skill_name
1	Python
3	SQL
8	Cloud Computing

Data integrity and constraints:

Primary keys, foreign keys, and constraints were used appropriately to ensure data consistency and performance.

Primary Keys:

Each table has a primary key to uniquely identify each record in the table:

- **Admin Table:** The primary key is admin_id, which uniquely identifies each administrator.
- **Applicant Table:** The primary key is applicant_id, which uniquely identifies each applicant.
- **Job_Posting Table:** The primary key is job_id, which uniquely identifies each job posting.
- **Application Table:** The primary key is application_id, which uniquely identifies each application.
- **Skills Table:** The primary key is skill_id, which uniquely identifies each skill.
- **Applicant_Skills Table:** This table uses a composite primary key consisting of applicant_id and skill_id, which together uniquely identify each skill assigned to an applicant.

Foreign Keys:

Ensures referential integrity (e.g., job_id in application references, job_id in job posting).
Foreign keys are used to establish relationships between tables, ensuring referential integrity.

- **Job_Posting Table**: The admin_id column in the Job_Posting table is a foreign key referencing the admin_id in the Admin table. This establishes a relationship between job postings and the admin who posted the job.
- **Application Table:** The job_id column in the Application table is a foreign key referencing the job_id in the Job_Posting table. This ensures that each application is linked to a specific job posting. The applicant_id column in the Application table is a foreign key referencing the applicant_id in the Applicant table. This ensures that each application is associated with a specific applicant.
- **Applicant_Skills Table:** The applicant_id column in the Applicant_Skills table is a foreign key referencing the applicant_id in the Applicant table. This establishes a relationship between applicants and the skills they possess. The skill_id column in the Applicant_Skills table is a foreign key referencing the skill_id in the Skills table. This ensures that each skill assigned to an applicant corresponds to an existing skill in the Skills table.
- **Unique Constraints:** Ensures that no two applicants can have the same email address.
- **Indexing:** Index frequently searched columns like job location and title to improve query performance.

Test Data:

Test data was inserted to verify functionality, including job postings, applicants, applications, and skills. Example test data is shown below:

```
-- Insert Admin records
INSERT INTO Admin (company_name, location, industry) VALUES
('Tech Solutions Inc.', 'New York, NY', 'Technology'),
('Healthcare Plus', 'Boston, MA', 'Healthcare'),
('Financial Experts Ltd', 'Chicago, IL', 'Finance'),
('Green Energy Co', 'San Francisco, CA', 'Energy');

-- Insert Applicant records
INSERT INTO Applicant (full_name, email, education, experience, resume_link, CV_link,
other_link, contact_number) VALUES
('John Doe', 'john.doe@email.com', 'Bachelor in Computer Science', '5 years software
development experience',
 'https://storage/resume/john_doe.pdf',                'https://storage/cv/john_doe.pdf',
'https://github.com/johndoe', '+1-234-567-8901'),
('Jane Smith', 'jane.smith@email.com', 'Master in Data Science', '3 years data analyst
experience',
 'https://storage/resume/jane_smith.pdf',            'https://storage/cv/jane_smith.pdf',
'https://linkedin.com/in/janesmith', '+1-234-567-8902'),
('Bob Wilson', 'bob.wilson@email.com', 'Bachelor in Business Administration', '4 years
project management experience',
 'https://storage/resume/bob_wilson.pdf',            'https://storage/cv/bob_wilson.pdf',
'https://portfolio.com/bobwilson', '+1-234-567-8903'),
('Alice Brown', 'alice.brown@email.com', 'Master in Artificial Intelligence', '2 years ML
engineer experience',
 'https://storage/resume/alice_brown.pdf',          'https://storage/cv/alice_brown.pdf',
'https://github.com/alicebrown', '+1-234-567-8904');

-- Insert Skills records
INSERT INTO Skills (skill_name) VALUES
('Python'),
('Java'),
('SQL'),
('Project Management'),
('Machine Learning'),
('Data Analysis'),
('JavaScript'),
('Cloud Computing');

-- Insert Applicant_Skills records
INSERT INTO Applicant_Skills (applicant_id, skill_id) VALUES
(1, 1), -- John Doe knows Python
(1, 3), -- John Doe knows SQL
(1, 7), -- John Doe knows JavaScript
(2, 1), -- Jane Smith knows Python
(2, 5), -- Jane Smith knows Machine Learning
(2, 6), -- Jane Smith knows Data Analysis
(3, 4), -- Bob Wilson knows Project Management
(3, 3), -- Bob Wilson knows SQL
(4, 1), -- Alice Brown knows Python
(4, 5), -- Alice Brown knows Machine Learning
(4, 8); -- Alice Brown knows Cloud Computing

-- Insert Job_Posting records
INSERT INTO Job_Posting (admin_id, title, description, location, salary_min, salary_max,
posted_date, skills_required, is_active) VALUES
```

```
(1, 'Senior Software Engineer', 'Looking for an experienced software engineer with strong
Python and JavaScript skills.',
 'New York, NY', 100000.00, 150000.00, '2024-01-15', 'Python, JavaScript, SQL', TRUE),
(1, 'Data Scientist', 'Seeking a data scientist with machine learning expertise.',
 'New York, NY', 90000.00, 130000.00, '2024-01-16', 'Python, Machine Learning, Data
Analysis', TRUE),
(2, 'Project Manager', 'Healthcare IT project manager needed for large-scale
implementations.',
 'Boston, MA', 85000.00, 120000.00, '2024-01-17', 'Project Management, Healthcare IT',
TRUE),
(3, 'ML Engineer', 'Looking for an ML engineer with cloud computing experience.',
 'Chicago, IL', 95000.00, 140000.00, '2024-01-18', 'Python, Machine Learning, Cloud
Computing', TRUE);

-- Insert Application records
INSERT INTO Application (job_id, applicant_id, application_date, status) VALUES
(1, 1, '2024-01-20', 'Reviewed'),
(2, 2, '2024-01-21', 'Reviewed'),
(3, 3, '2024-01-22', 'Reviewed'),
(4, 4, '2024-01-23', 'Reviewed'),
(1, 2, '2024-01-24', 'Reviewed'),
(2, 4, '2024-01-25', 'Reviewed');

-- Verify data insertion
SELECT 'Admin count: ' as 'Table Check', COUNT(*) FROM Admin
UNION ALL
SELECT 'Applicant count: ', COUNT(*) FROM Applicant
UNION ALL
SELECT 'Skills count: ', COUNT(*) FROM Skills
UNION ALL
SELECT 'Applicant_Skills count: ', COUNT(*) FROM Applicant_Skills
UNION ALL
SELECT 'Job_Posting count: ', COUNT(*) FROM Job_Posting
UNION ALL
SELECT 'Application count: ', COUNT(*) FROM Application;
```

✓	18	17:10:10	INSERT INTO Admin (company_name, location, industry) VALUES ('Tech Solutions Inc.', 'New York, NY', 'Technology'), ('Healthcare Plus', 'Boston, MA', 'Healthcare'), ('Financial Experts Ltd', 'Chicago, IL', 'Fi...	4 row(s) affected Records: 4
✓	19	17:10:10	INSERT INTO Applicant (full_name, email, education, experience, resume_link, CV_link, other_link, contact_number) VALUES ('John Doe', 'john.doe@email.com', 'Bachelor in Computer Science', '5 years soft...	4 row(s) affected Records: 4
✓	20	17:10:10	INSERT INTO Skills (skill_name) VALUES ('Python'), ('Java'), ('SQL'), ('Project Management'), ('Machine Learning'), ('Data Analysis'), ('JavaScript'), ('Cloud Computing')	8 row(s) affected Records: 8
✓	21	17:10:10	INSERT INTO Applicant_Skills (applicant_id, skill_id) VALUES (1, 1), -- John Doe knows Python (1, 3), -- John Doe knows SQL (1, 7), -- John Doe knows JavaScript (2, 1), -- Jane Smith knows Python (2, 5), -...	11 row(s) affected Records: 11
✓	22	17:10:10	INSERT INTO Job_Posting (admin_id, title, description, location, salary_min, salary_max, posted_date, skills_required, is_active) VALUES (1, 'Senior Software Engineer', 'Looking for an experienced software...	4 row(s) affected Records: 4
✗	23	17:10:10	INSERT INTO Application (job_id, applicant_id, application_date, status) VALUES (1, 1, '2024-01-20', 'pending'), (2, 2, '2024-01-21', 'reviewed'), (3, 3, '2024-01-22', 'shortlisted'), (4, 4, '2024-01-23', 'pendi...	Error Code: 1265. Data trunca
✓	24	17:10:41	INSERT INTO Application (job_id, applicant_id, application_date, status) VALUES (1, 1, '2024-01-20', 'Reviewed'), (2, 2, '2024-01-21', 'Reviewed'), (3, 3, '2024-01-22', 'Reviewed'), (4, 4, '2024-01-23', 'Revi...	6 row(s) affected Records: 6

4. Application Integration:

In this project, the database is integrated with a front-end interface developed using HTML, CSS, JavaScript, and React. The integration facilitates a seamless user experience by allowing users to interact with the database through a web interface, performing essential operations such as creating, reading, updating, and deleting (CRUD) data.

User Interface Implementation:

The front-end user interface is built using Tailwind CSS, APIs and JavaScript, with React as the primary JavaScript library for building dynamic and interactive UI components. The React framework was chosen to ensure a responsive and smooth user experience, especially when dealing with real-time updates and state management.

1. **React:**
   React was used to create reusable components and manage the state of the application. React components were used for rendering dynamic content such as lists of job postings, applicant applications, and skill sets. Additionally, React's state management helps in managing form submissions, filtering job postings, and updating application statuses.
2. **Tailwind CSS**
   Our project utilized Tailwind CSS, a utility-first CSS framework, for styling the front-end components. Tailwind was chosen due to its flexibility and ability to streamline the development process by providing pre-defined utility classes for common CSS functionalities.

Functionality:

The application supports several key functionalities, including:

- **Displaying Job Postings:** The UI fetches data from the database and displays all active job postings with details such as job title, location, company name, and required skills.
- **Searching and Filtering Jobs:** Users can search for jobs by title, company, location, or required skills. React is used to handle user input dynamically and update the display of job listings accordingly.
- **Applying for Jobs:** Applicants can apply for jobs by submitting an application form. The data entered in the form is captured and sent to the backend, where it is inserted into the database.
- **Updating Application Status:** Admins can update the status of an applicant's job application through a simple interface that allows the admin to mark applications as "reviewed," "interview scheduled," or other relevant statuses.
- **Managing Applicants and Jobs:** Admin users can also perform CRUD operations on job postings and applicant profiles, such as adding new job listings or editing applicant information.

Tools and Frameworks Used:

- **React:** provides a modern framework to build responsive, dynamic, and modular components.
- **Tailwind CSS:** Used for the overall structure and styling of the website.
- **JavaScript:** Handles dynamic functionalities such as form handling, search, and filtering.
- **Backend API:** The backend API, integrated with MySQL, serves as a bridge between the frontend and the database, ensuring smooth data interaction. It is responsible for receiving requests from the React front-end, executing SQL queries on the MySQL database, and returning the results in a structured format (e.g., JSON). Key functionalities include retrieving job postings, managing applicant data, and handling applications
- **MYSQL:** The project utilized MySQL, a widely used relational database management system (RDBMS), to handle and manage data efficiently. MySQL was chosen for its robustness, scalability, and ease of integration with the back-end technologies used in the project (Node.js/Express, for instance).

Screenshots:

Home Page:

The home page serves as the entry point for the application, offering two distinct options for users:

- **Employers:** Redirects to the Employer Dashboard, where employers can manage job postings and view applications.
- **Job Seekers:** Redirects to the Applicant Dashboard, where applicants can search for jobs, manage applications, and update their profiles.

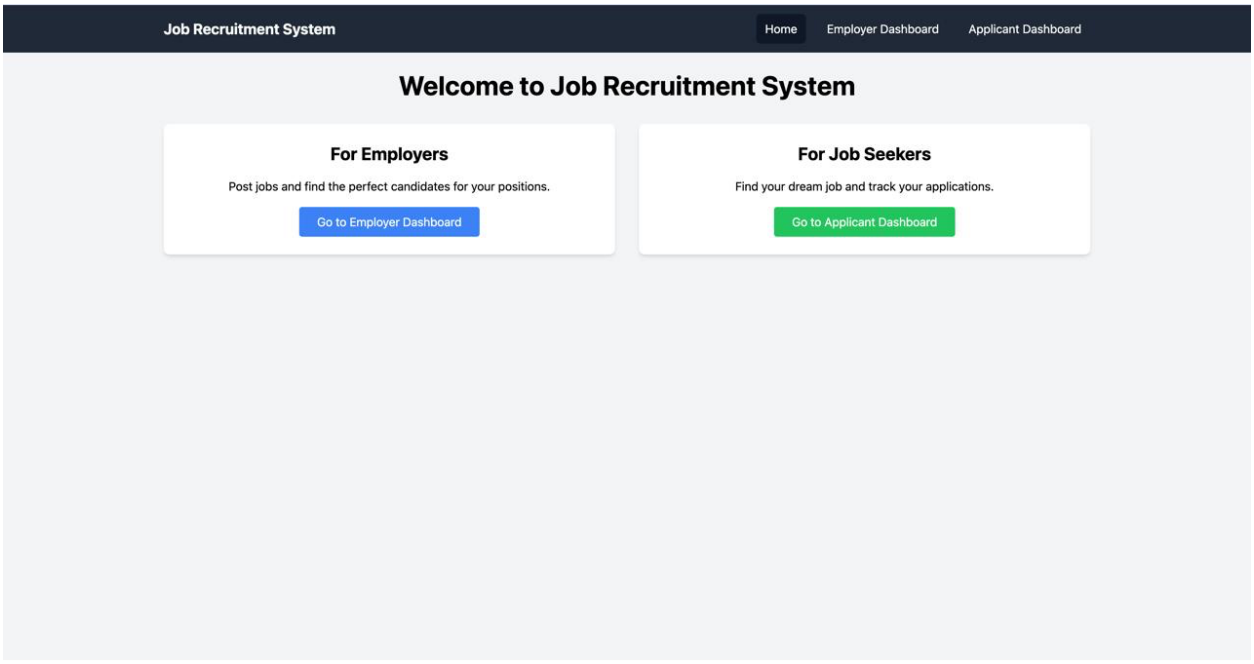


Fig. 1: JRS Home Page

Employer Dashboard

The Employer Dashboard provides a comprehensive overview for employers. Key features include:

- **Active Jobs:** Displays the total number of active job postings.
- **Total Applicants:** Shows the total number of applicants.
- **New Applicants:** Indicates the number of new applicants.
- **Shortlisted Applicants:** Displays the count of shortlisted candidates.

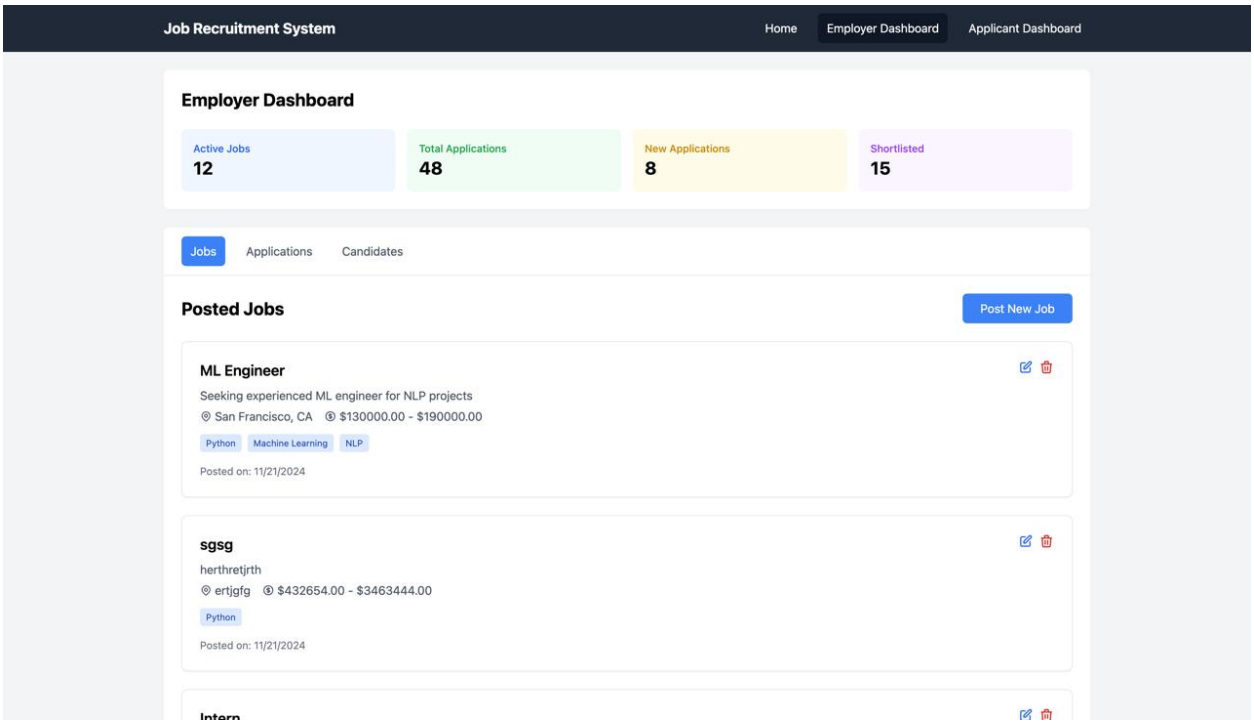


Fig 2: Employer Dashboard

Additional sections include:

1. **Job Section:**
   Employers can:
   - Post new job listings.
   - Edit existing job postings.
   - Delete job postings.
   - View all active job postings.

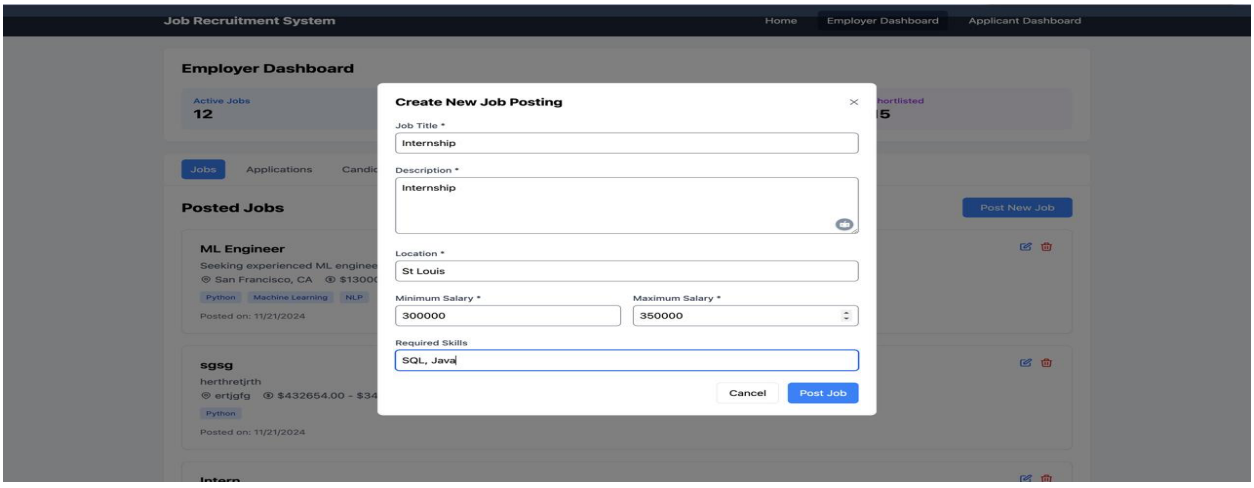


Fig 3: Creating New Job Posting

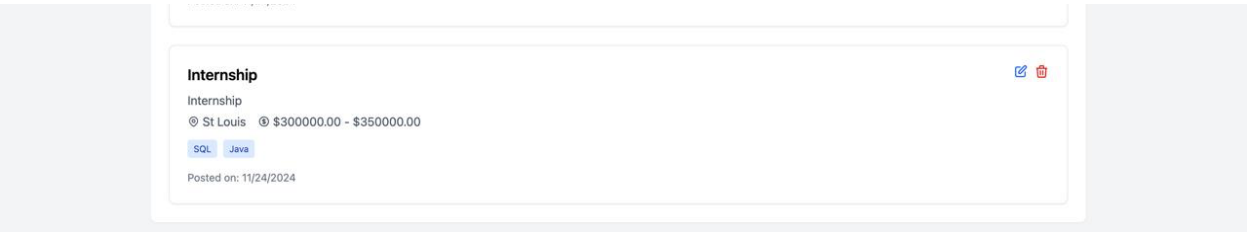


Fig 4: Successfully added the job title internship

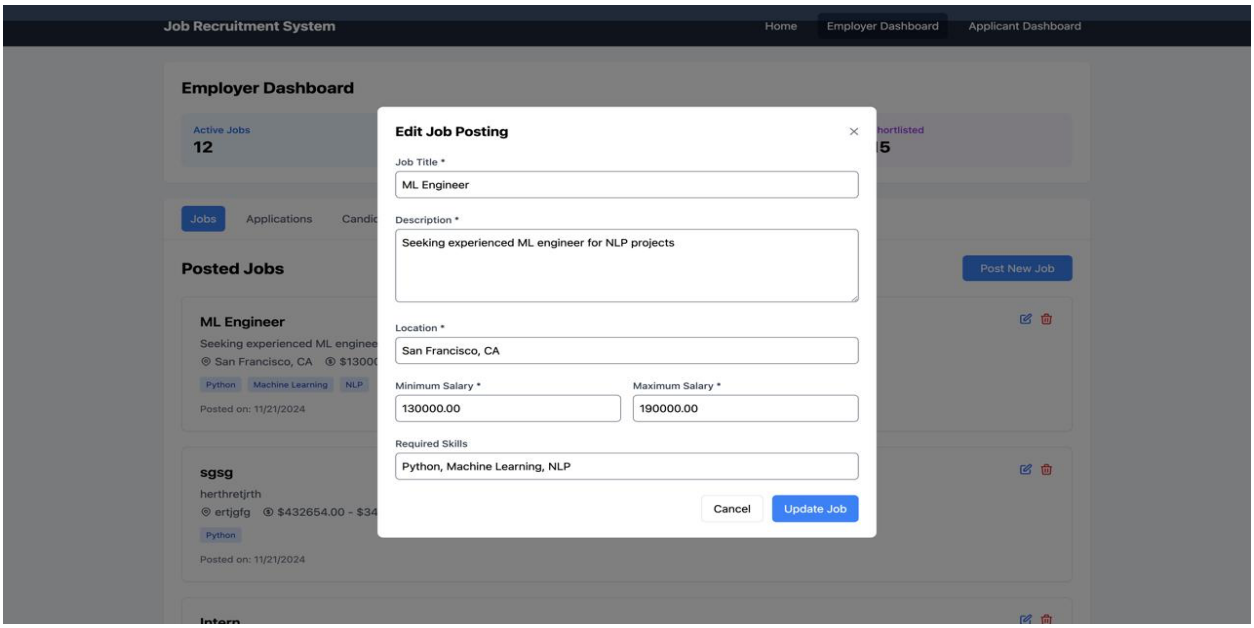


Fig 5: Editing the Job Posting

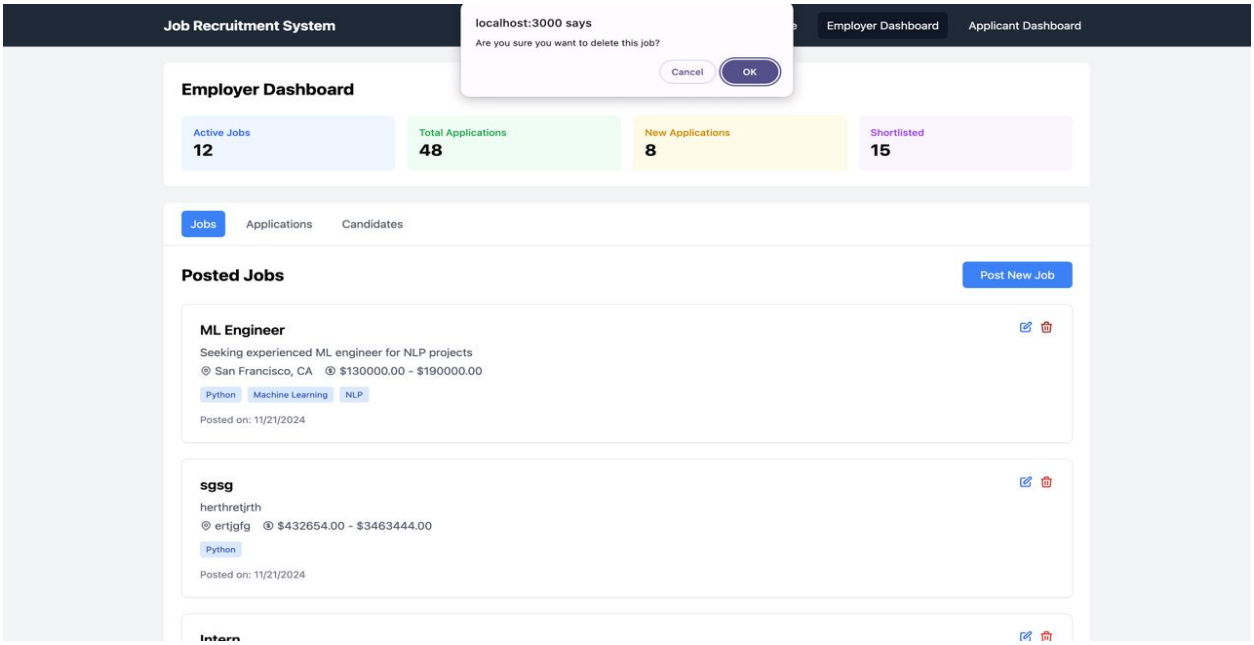


Fig 6: Deleting the Job

2. **Applicant Section:**
   Employers can view an overview of applications submitted for their job postings.

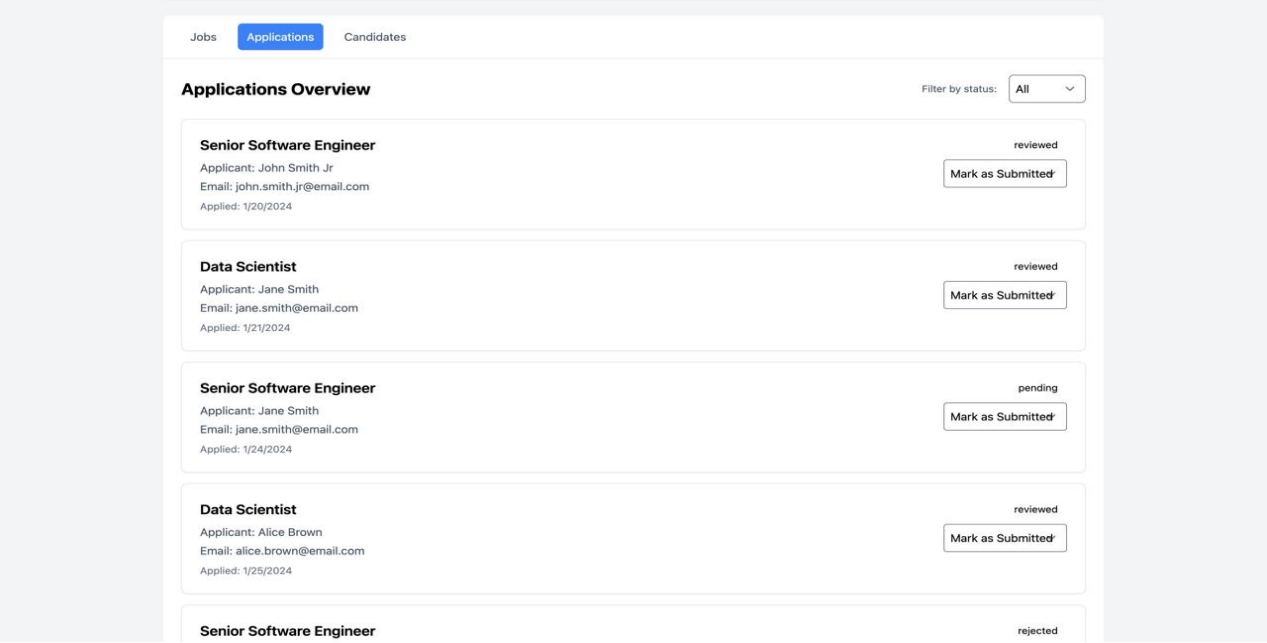


Fig 7: Application Overview

3. **Candidates Section:**
   Employers can:
   - Search for candidates.
   - Sort candidates by name.
   - Filter candidates by skills.

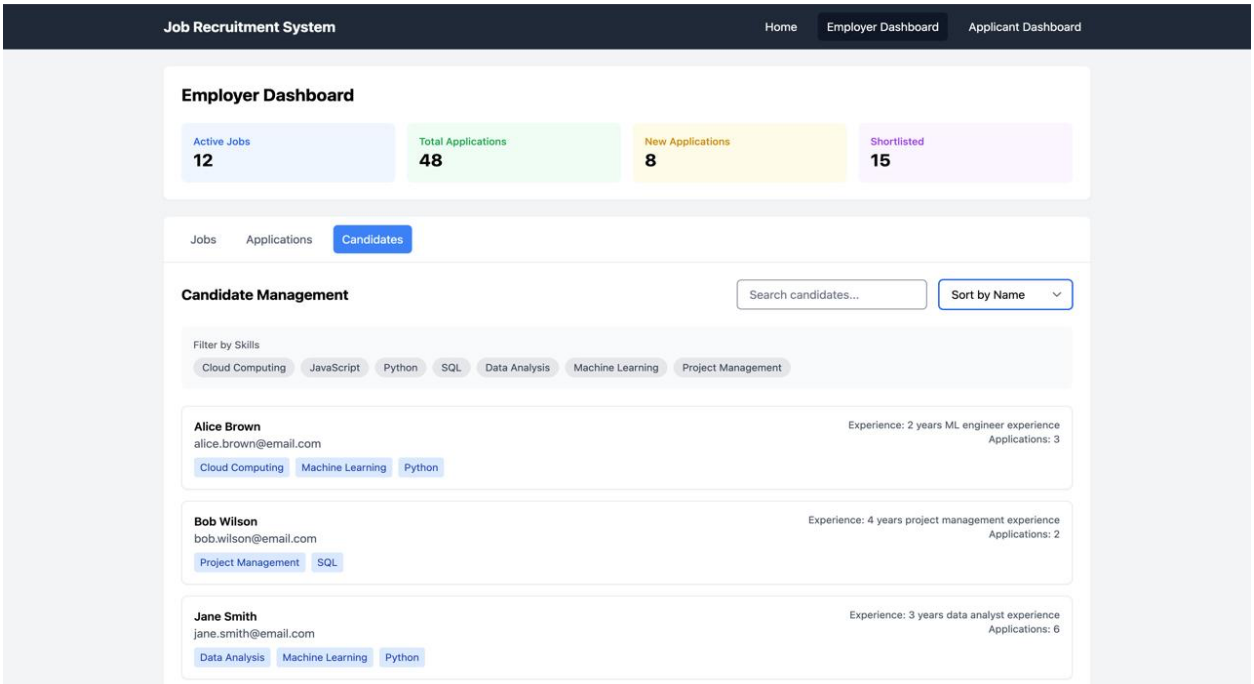


Fig 8: Candidate Section

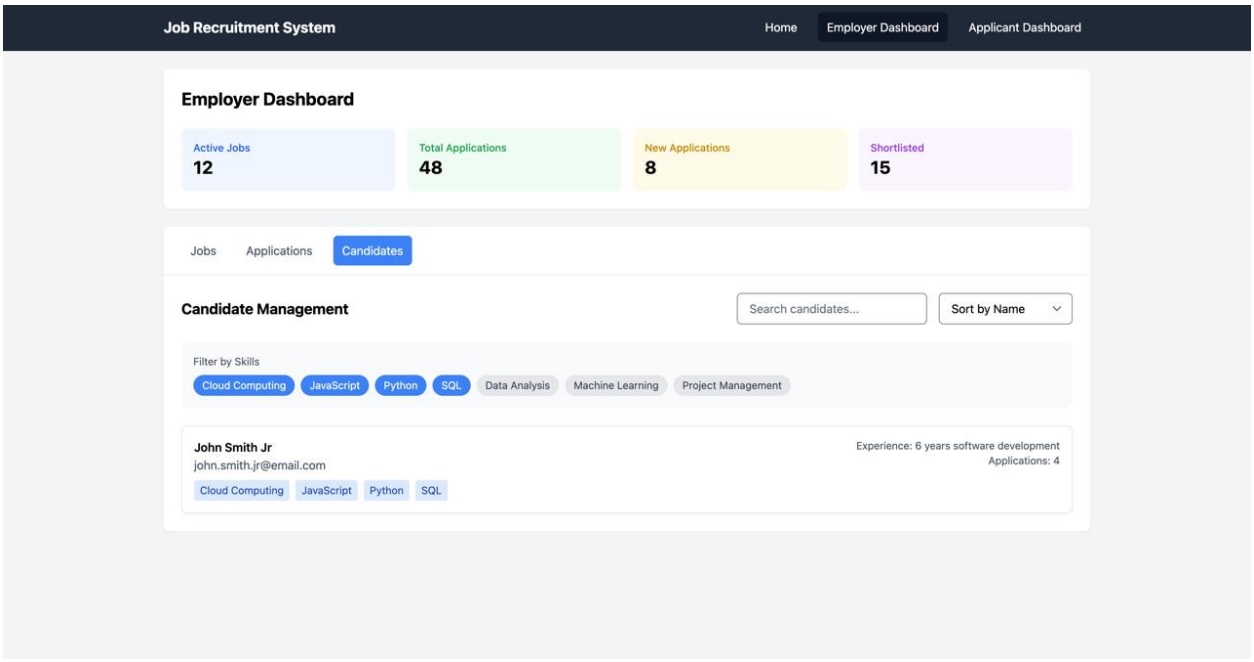


Fig 9: Filter by skills

Applicant Dashboard

The Applicant Dashboard offers tailored functionalities for job seekers. It consists fo three main sections:

1. **Job Search:**

   Applicants can search for jobs using:

   - Keywords (job title or skills)
   - Location
   - Industry
   - Sort options (e.g., most recent postings). Once a suitable job is found , the applicant can apply directly through the platform.
   - Apply for the job.

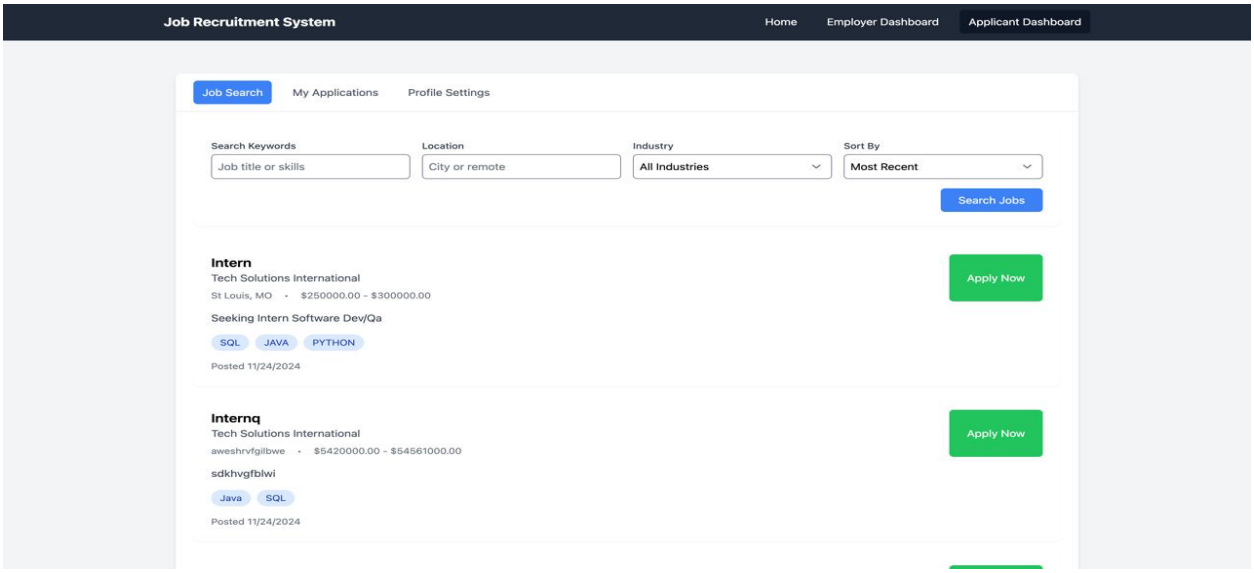


Fig 10 : Job search

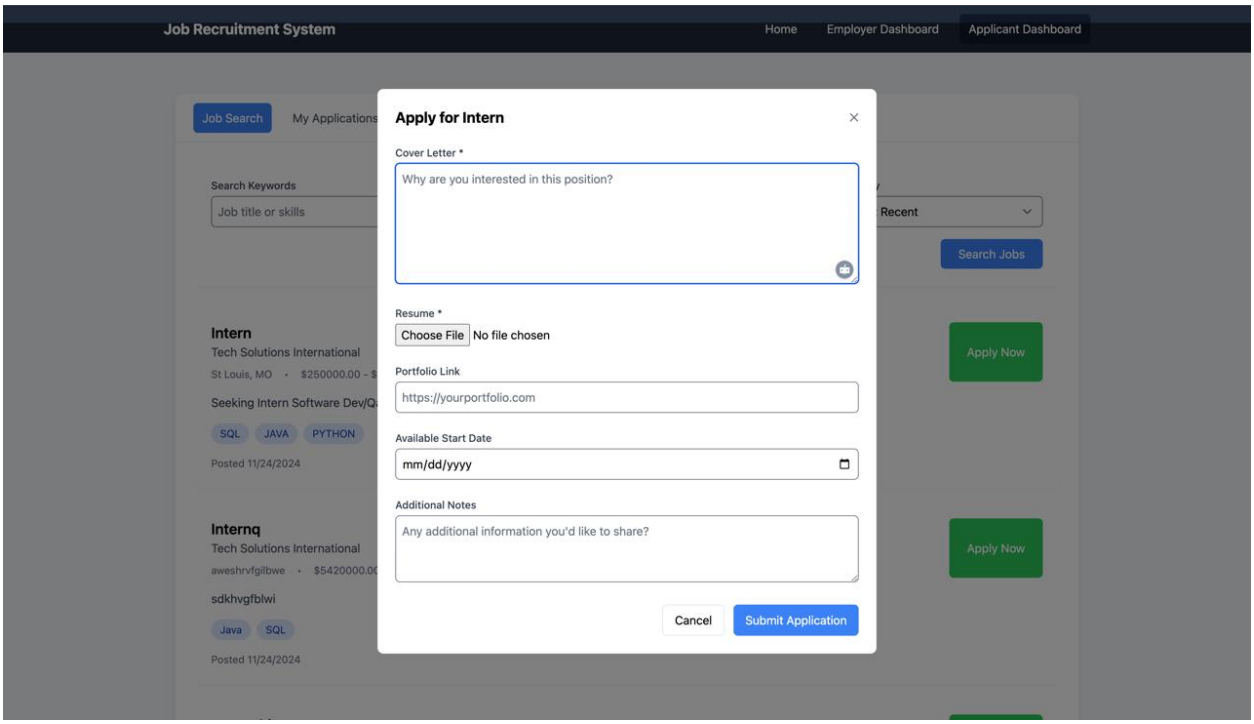


Fig 11: Applicant applying for the job.

2. My Applications:

This section allows applicants to:

- View all their submitted applications.
- Filter applications by status (e.g., pending, reviewed).

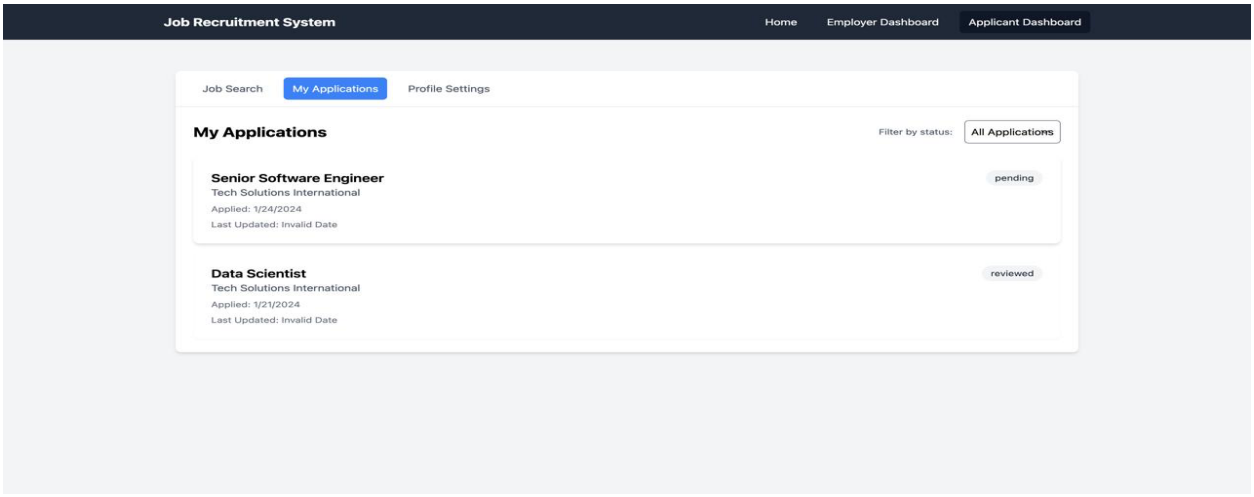


Fig 12: My Applications

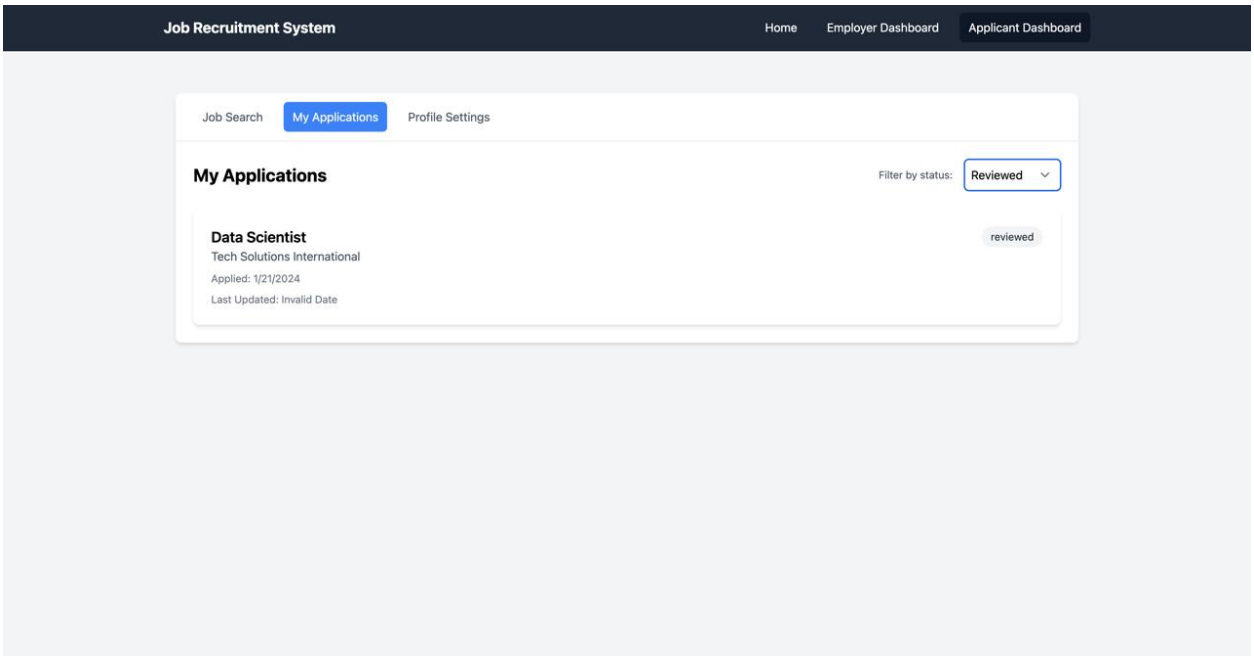


Fig 13: Filter by reviewed

3. Profile Settings:

Applicants can update their profile settings, such as personal details, skills, and preferences.

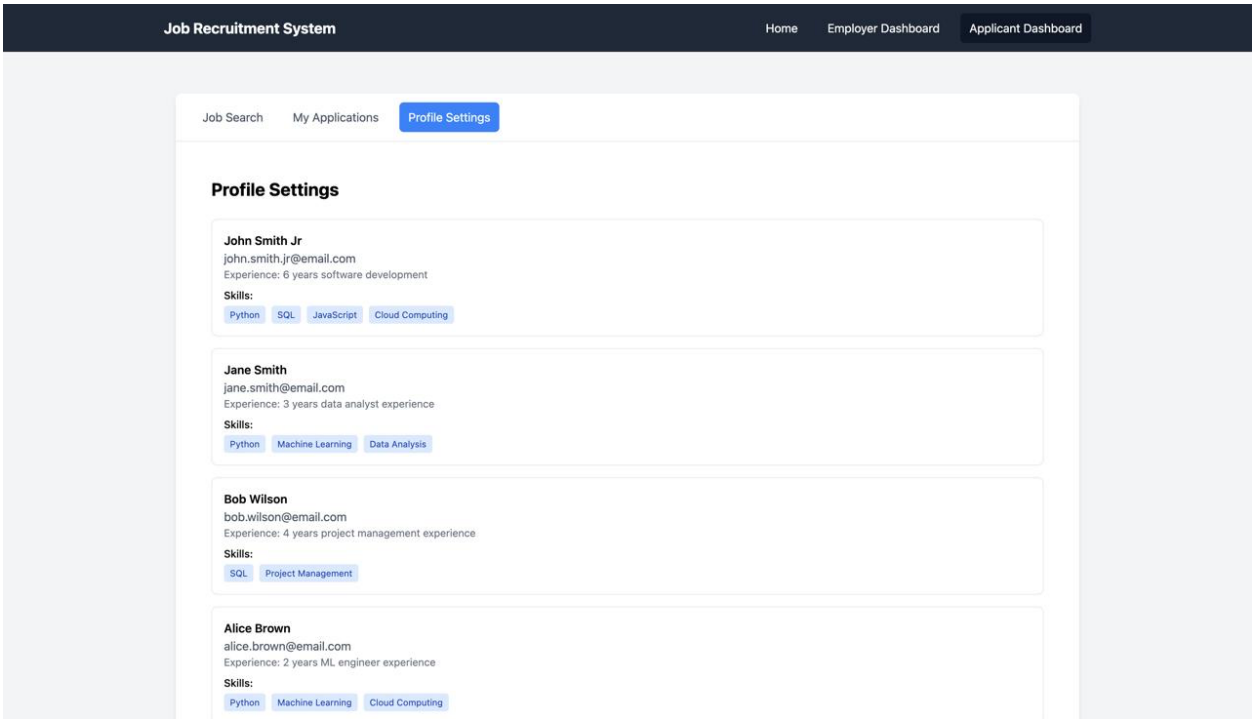


Fig 14: Applicant Profile Settings.

5. Teamwork and Collaboration

Team Contribution:

Each team member contributed significantly to different aspects of the project. Below is a summary of each member's contributions:

1. **Vigneshwar Reddy Gangireddy [Team Lead]:**
   Effectively led the team, overseeing the project from start to finish. His work in designing the ERD, writing SQL queries, and ensuring all tasks were aligned with project goals was crucial. He demonstrated great leadership and kept the team on track.

2. **Alex Chilaka**
   Took full responsibility for the frontend, ensuring the interface was both attractive and functional. His work in React was vital for creating an interactive and dynamic user experience. He was proactive in ensuring the application was responsive.

3. **Sumanth Kothamasu**
   I was instrumental in setting up the backend. They handled API development and the logic for data interaction between the frontend and the database. Their problem-solving skills were helpful when debugging critical features.

4. **Bharath Bandlamudi**
   Played a significant role in managing the integration between the frontend and backend. They ensured that the React components communicated effectively with the API and database. They contributed to resolving data flow issues and optimizing the performance of certain features, such as filtering and sorting candidates.

5. **Kasi Viswanadh Mogali**
   Contributed to both the frontend and backend, assisting in creating React components and ensuring they functioned correctly with the backend API. They worked on specific features like job search filters and profile settings. Additionally, they were responsible for documentation and reporting, ensuring that technical specifications, implementation details, and user instructions were clear and well-structured.

Peer Evaluation (On 5):

Criteria	Vigneshwar Reddy Gangireddy	Alex Chilaka	Sumanth Kothamasu	Bharath Bandlamudi	Kasi Viswanadh Mogali
1. Contributed Meaningfully to the Project	5	5	5	5	5
2. Quality of Work	5	5	5	5	5
3. Communication and Collaboration	5	5	5	5	5
4. Technical Expertise	5	5	5	5	5
5. Timeliness and Meeting Deadlines	5	5	5	5	5
6. Responsibility and Initiative	5	5	5	5	5