SIMCA
BREEDING
THOUGHTS

A

PROJECT REPORT ON

"JOB-LINK JOB PORTAL"

**SUBMITTED
TO**

SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

FOR THE PARTIAL FULFILLMENT OF

MASTER OF COMPUTER APPLICATION

(MCA-I, SEM.-II)

BY

KOKANE VISHAL DATTATRAY

UNDER THE GUIDANCE OF

PROF. YOGESH SHARMA

THROUGH

**THE DIRECTOR
SINHGAD INSTITUTE OF MANAGEMENT AND COMPUTER APPLICATION
(SIMCA), NARHE, PUNE (AY. 2022-2023)**

SINHGAD TECHNICAL EDUCATION SOCIETY'S

SINHGAD INSTITUTE OF
MANAGEMENT & COMPUTER APPLICATION

(Affiliated to Savitribai Phule Pune University & Approved by AICTE)

'NAAC' Accredited with 'A' Grade

S. No. 49/1, Off Westernly Bypass, Pune – Mumbai Expressway, Narhe, Pune – 411041, Tel : (020) 66831896 / 66831908 / 66831907
E-mail : director_mca_simca@sinhgad.edu Website : www.sinhgad.edu

Sinhgad Institutes

SIMCA
BREEDING THOUGHTS

Prof. M. N. Navale	**Dr. (Mrs.) Sunanda M. Navale**	**Dr. Vijaya Puranik**
M.E. (Elec), MIE, MBA	B.A., P.P.M., Ph.D.	*M.Sc., MMM,MPM, Ph.D.*
FOUNDER PRESIDENT		FOUNDER SECRETARY
DIRECTOR, SIMCA		

CERTIFICATE

This is to certify that, the project entitled "***Job-Link Job Portal*"**, being submitted for the partial fulfillment of the degree of **Master of Computer Application** by him to **Sinhgad Institute of Management and Computer Application affiliated to Savitribai Phule Pune University, Pune** is the result of the original work completed by ***Kokane Vishal Dattatray*** under the guidance of ***Prof. Yogesh Sharma***.

To the best of our knowledge and belief, this work has not been previously submitted by the award of any degree or diploma of Savitribai Phule Pune University or any other University.

Place: Narhe, Pune

Date:

Prof. Yogesh Sharma	**Prof. Navanath Choudhari**	**Dr. Vijaya Puranik**
Internal Guide	**Project Co-Ordinator**	**Director SIMCA**

External Examiner

DECLARATION

I, the undersigned hereby declare that the project titled "**Job-Link Job Portal", being** submitted for the award of degree of **Master of Computer Application** by me to **Sinhgad Institute of Management and Computer Application (SIMCA) affiliated to Savitribai Phule Pune University is** the result of an independent work carried out under the guidance of **Prof**. **Yogesh Sharma,** is my original work. Further I declare that this project has not been submitted to this or any Institution for the award of any degree.

Place : Pune   Kokane Vishal Dattatray

Date :   (Student)

ACKNOWLEDGEMENT

We take this opportunity to express our gratitude towards all those who have helped us throughout the successful completion of our project Firstly, we would like to thank our Director **Dr. Vijaya Puranik**, and project coordinator **Prof. Navanath Choudhari** of Masters in Computer Application, for permitting us to complete our project and for showing faith in us and allowing us to develop **Job-Link Job Portal**. We must convey our gratitude to our guide **Prof. Yogesh Sharma** for giving us the constant guidance of inspiration and help in preparing the project, personally, correcting our work and providing encouragement throughout the project

Kokane Vishal Dattatray

Student Name

INDEX

1. Introduction

In today's rapidly evolving job market, connecting job seekers with suitable employment opportunities has become increasingly challenging. With the growing reliance on digital platforms, there is a need for an efficient and user-friendly online job portal that simplifies the job search process for both candidates and employers. Introducing Job-Link, a comprehensive online job portal designed to bridge the gap between talent and opportunities.

The main objective of Job-Link is to provide a comprehensive database of job listings across various industries and job categories. Job seekers can create profiles, upload their resumes, and search for relevant job opportunities based on their skills, qualifications, and preferences. The portal offers search filters and algorithms to help job seekers find the most suitable positions.

Employers, on the other hand, can create company profiles and post job openings on Job-Link. They can specify the desired qualifications and job requirements, and the platform will match these criteria with potential candidates from the job seeker database. Employers can also actively search for candidates and directly contact them on preferences.

Overall, Job-Link aims to bridge the gap between job seekers and employers by offering a centralized platform that simplifies the job search and recruitment process, ultimately helping individuals find suitable employment opportunities and employers find qualified candidates.

1.1 Abstract

The online job portal project, Job-Link, is a platform that connects job seekers with potential employers through a web-based interface. It simplifies the job search and recruitment process by providing a centralized database of job listings from various industries and job categories.

The Job-Link portal provides a user-friendly interface, making it easy for job seekers and employers to navigate and interact with the platform. The goal of Job-Link is to streamline the job search and recruitment process, providing a convenient and efficient solution for individuals seeking employment and employers looking for qualified candidates.

By leveraging technology and offering a comprehensive range of features, Job-Link aims to bridge the gap between job seekers and employers, ultimately facilitating successful job placements and career opportunities for individuals in need of employment.

1.2 Existing System and Need for System

The existing job search process often involves traditional methods such as newspaper advertisements, physical job boards, and networking. Job seekers have to manually search for job openings, submit physical resumes, and rely on personal connections for job opportunities. Similarly, employers have to manage a large number of resumes, conduct manual screenings, and advertise job openings through limited channels.

The Need for the System:

- Centralized Platform
- Access to a Wider Job Market
- Efficient Job Search
- Streamlined Recruitment Process
- Enhanced Connectivity
- Additional Resources and Support

2. Proposed System

The proposed system, Job-Link, is an online job portal that aims to revolutionize the job search and recruitment process by leveraging technology and providing a user-friendly platform for job seekers and employers. The system offers several key features and improvements over the existing system**:**

- User-Friendly Interface
- Advanced Search and Matching Algorithms
- Enhanced Communication and Connectivity
- Job Alerts and Notifications
- Ads free experience
- Additional Resources and Support

2.1 Objectives of Proposed System

- Streamline the Job Search Process: The primary objective of Job-Link is to simplify and streamline the job search process for job seekers. The system will provide advanced search filters and matching algorithms to help job seekers find relevant job opportunities based on their skills, qualifications, and preferences.

- Connect Job Seekers with Employers: Job-Link aims to connect job seekers with potential employers by providing a centralized platform. The system will facilitate direct communication and interaction between job seekers and employers, enabling efficient and seamless connections.

- Increase Access to Job Opportunities: The proposed system seeks to expand job seekers' access to a wider range of job opportunities. Job-Link will aggregate job listings from various industries and job categories, providing a comprehensive database of job openings for job seekers to explore.

- Improve Recruitment Efficiency: Job-Link aims to enhance the efficiency of the recruitment process for employers. The system will allow employers to post job openings, specify requirements, and access a pool of pre-screened candidates, thereby reducing the time and effort involved in finding qualified candidates.

2.2 Users Summery

The proposed Job-Link system targets two primary user groups: job seekers and employers. Here's a summary of how each user group benefits from the system:

Job Seekers:

- Job seekers can create profiles on Job-Link, showcasing their skills, qualifications, and experience.
- They can search for relevant job opportunities using advanced search filters.
- The system provides access to a wide range of job listings from various industries and job categories.
- Job-Link facilitates direct communication with employers, increasing the chances of being noticed by potential employers and securing suitable employment.

Employers:

- Employers can create company profiles on Job-Link, providing information about their organization and culture.
- They can post job openings with specific requirements and qualifications.
- The system offers access to a large pool of pre-screened candidates, saving time and effort in the recruitment process.
- Employers can search for potential candidates based on their qualifications and directly contact face-to-face.

2.3 Scope of the system Requirements

Functional Requirements of Job-Link:

- User Registration and Profile Management:

Allow job seekers and employers to register accounts.

Provide options for creating and editing user profiles with relevant information.

Enable users to manage their account settings, including password changes and email preferences.

- Job Search and Filtering:

Implement search functionality for job seekers to find relevant job openings.

Include filters for refining search results based on industry, location, salary, experience level, and other criteria.

Display search results in a clear and organized manner.

- Job Posting and Management:

Enable employers to post job openings, including job titles, descriptions, requirements, and application instructions.

Provide options for editing, updating, and removing job postings as needed.

Allow employers to manage and track applications received for each job posting.

- Resume and Application Management:

Allow job seekers to upload their resumes, cover letters, and other supporting documents. Provide options for managing and updating resumes and applications.

Enable employers to review, rate, and sort received applications.

Non-Functional Requirements of Job-Link:

- Usability:

Ensure a user-friendly interface with intuitive navigation and clear instructions.

Provide responsive design for seamless access on various devices and screen sizes.

- Performance:

Optimize system performance to handle a large volume of users and concurrent job searches. Ensure fast response times for search queries and page loading.

- Security:

Implement robust security measures to protect user data and prevent unauthorized access. Employ secure authentication and authorization mechanisms.

- Reliability:

Minimize system downtime and errors to ensure reliable access.

Implement backup and recovery mechanisms to safeguard data in case of failures.

- Compatibility:

Support popular web browsers and ensure compatibility across different platforms.

Adhere to web standards and accessibility guidelines for users with disabilities.

- Performance:

Provide appropriate error handling and error messaging to guide users in case of system issues or input errors.

2.4 System requirements

2.4.1 Software Requirements

- Technology: Python-Django, Js, HTML, CSS.
- Web Browser (Chrome)
- Visual Studio Code
- SQLite Database

2.4.2 Hardware Requirements

- Processor: Intel/AMD dual-core or above
- RAM: 4 GB RAM
- Disk: 10 GB of SSD/HDD
- Operating system: Linux, Windows

3. Requirement determination and Analysis

3.1 Fact Finding methods

- Interviews: Interviews involve face-to-face discussions with users, and subject matter experts to gather information about their needs and expectations. These interviews can be structured or unstructured and can be conducted in person or remotely.

- Observation: Observation involves watching users as they interact with the current system or with prototypes to gather information about their behavior, preferences, and pain points. This can be done in a lab or in the field, depending on the needs of the project.

- Document analysis: Document analysis involves reviewing existing documentation, such as user manuals and technical specifications, to identify requirements and constraints that may have been overlooked or not clearly communicated.

- Focus groups: Focus groups involve bringing together a small group of users to discuss their needs and expectations in a group setting. This can be used to gather both quantitative and qualitative data and to identify common themes and issues.

- Prototyping: Prototyping involves creating a simple, functional model of the website to test with users. This can be used to gather feedback and identify areas for improvement before the final system is developed.

- Brainstorming: Brainstorming involves generating ideas and solutions through a collaborative and creative process with subject matter experts. This can be used to identify new requirements and ideas that may not have been considered previously.

3.2 Feasibility study

- Technical Feasibility:

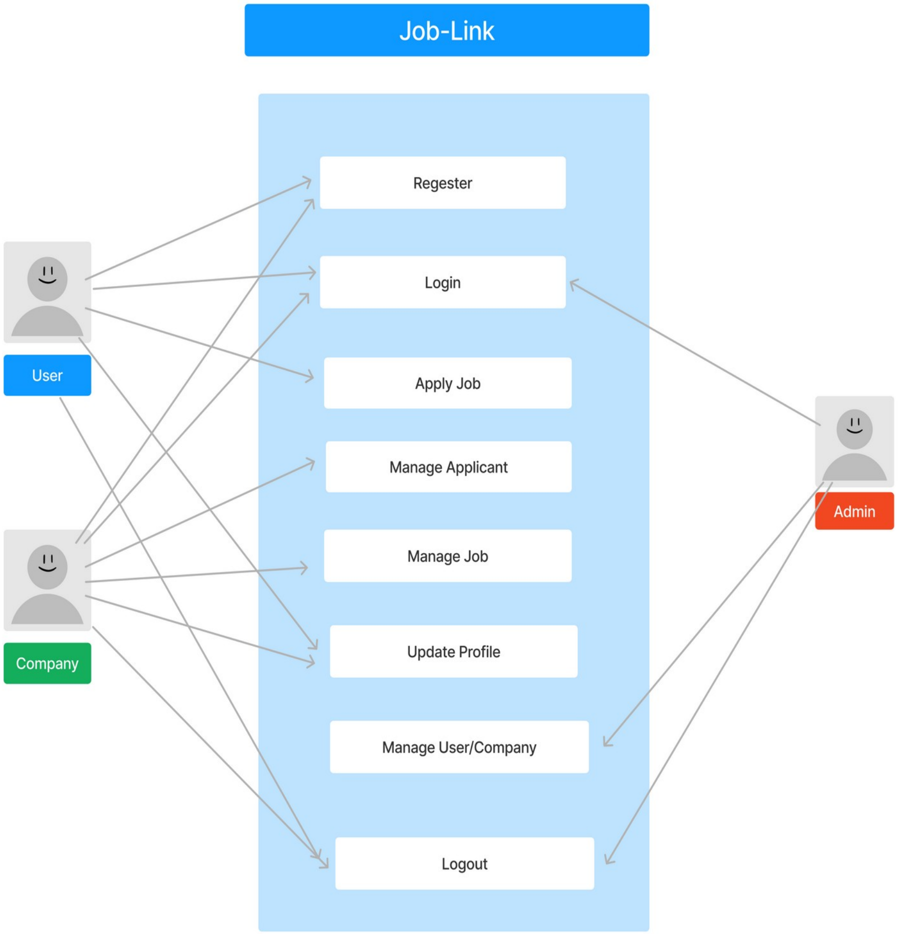
    - Availability of Technology: Assessing if the required hardware, software, and networking infrastructure needed for the Job-Link system are readily available or can be obtained within the project's constraints.
    - Skilled Resources: Determining if there are skilled developers, designers, and system administrators with the expertise needed to build and maintain the Job-Link system.
    - Integration Capability: Evaluating if the proposed system can integrate smoothly with existing systems, databases, and platforms used by job seekers and employers without significant technical difficulties.
    - Security and Privacy: Verifying that appropriate security measures can be implemented to protect user data, prevent unauthorized access, and maintain privacy and confidentiality within the Job-Link system.

- Operational Feasibility:

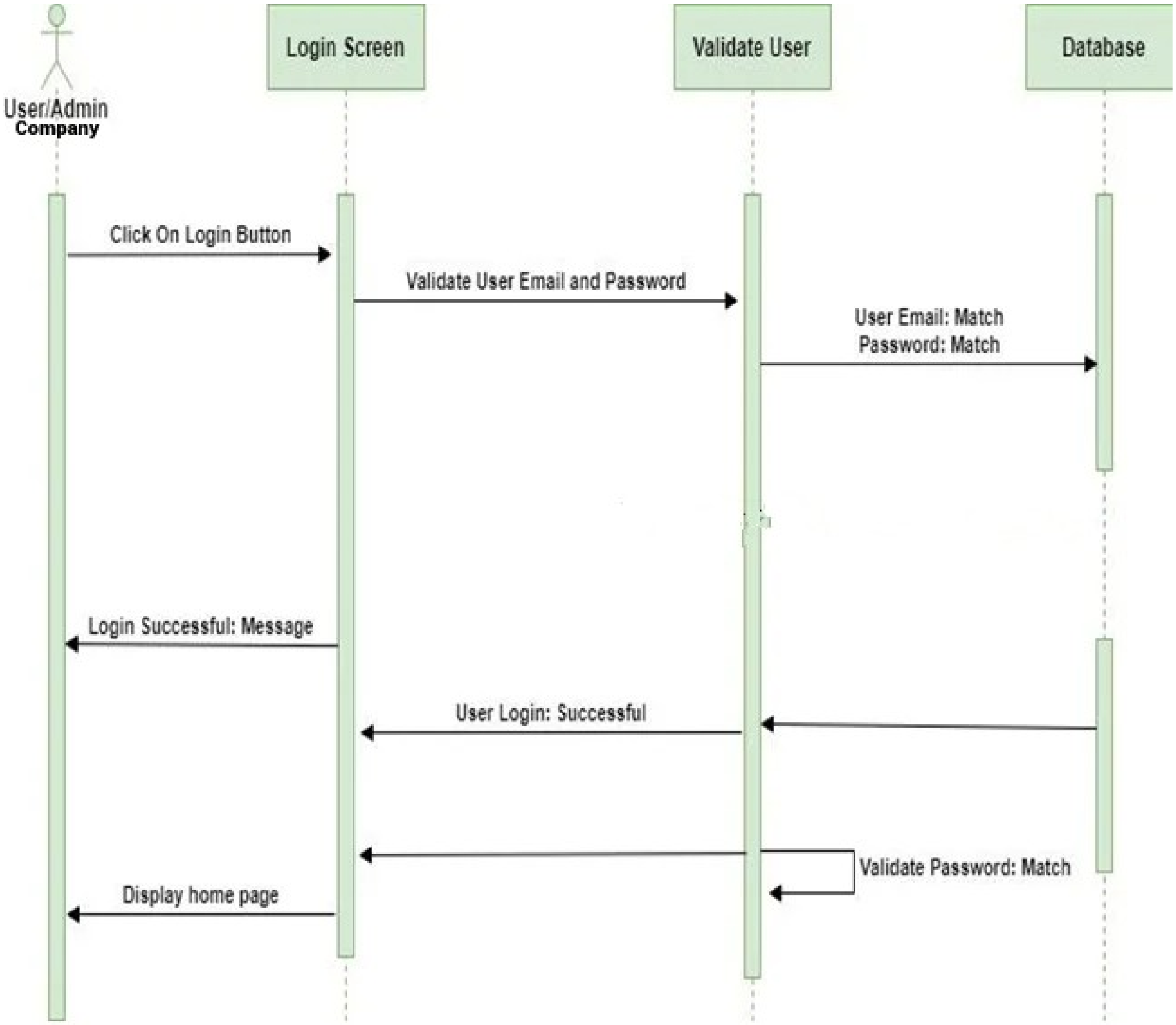
    - User Acceptance: Assessing the willingness and acceptance of potential users, including job seekers and employers, to adopt and use the Job-Link system effectively.
    - Organizational Impact: Evaluating the potential impact of implementing the system on the organization's processes, workflows, and resources, and assessing the organization's readiness for change.
    - Legal and Regulatory Compliance: Ensuring that the Job-Link system complies with applicable laws, regulations, and industry standards related to data protection, privacy, equal employment opportunity, and other relevant requirements.

4. System Analysis and Design

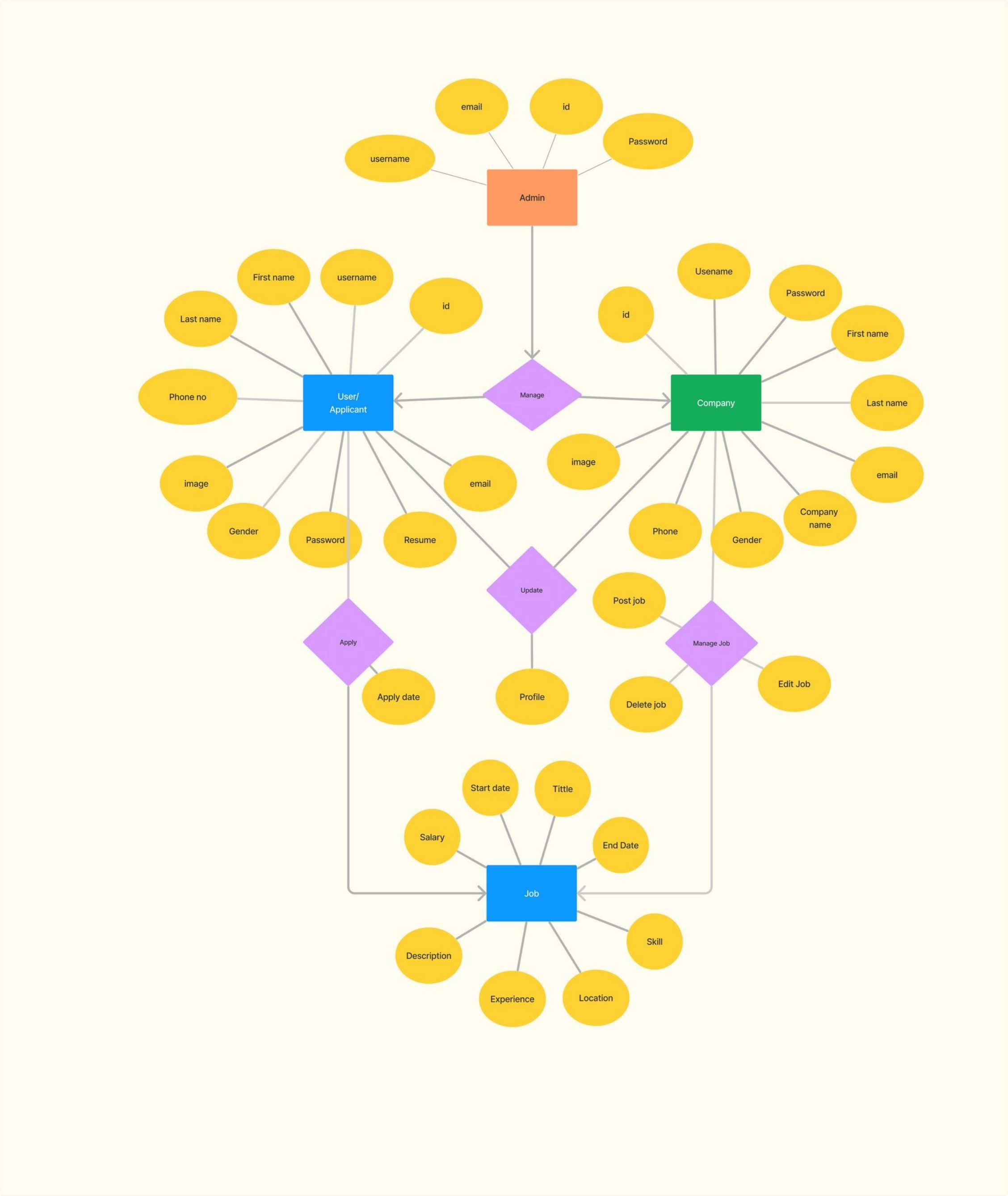
4.1 Use Case Diagram



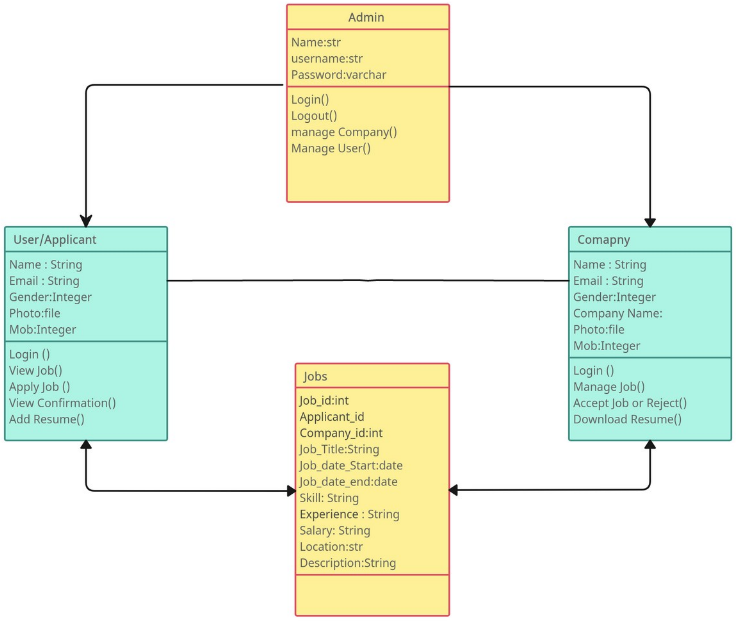
4.2 Sequence Diagram



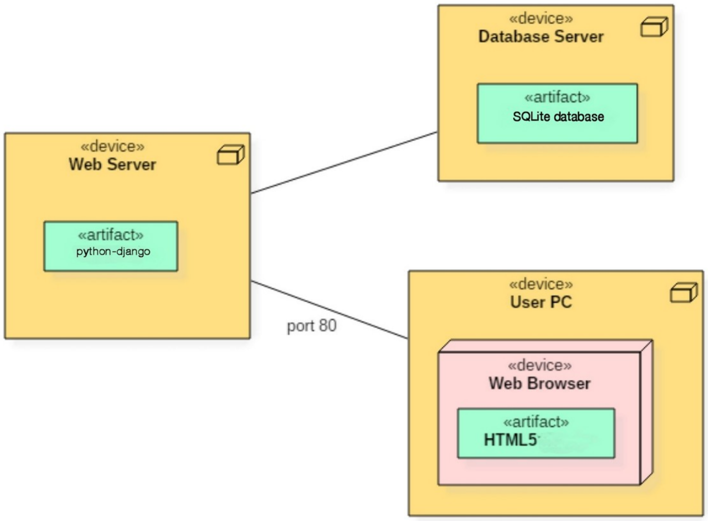
4.3 ER-D


email
id
username
Password
Admin
First name
username
Last name
id
Usename
Password
id
First name
Phone no
User/
Applicant
Manage
Company
Last name
image
email
image
email
Gender
Password
Resume
Phone
Gender
Company
name
Update
Post job
Apply
Manage Job
Apply date
Profile
Delete job
Edit Job
Start date
Tittle
Salary
End Date
Job
Skill
Description
Experience
Location

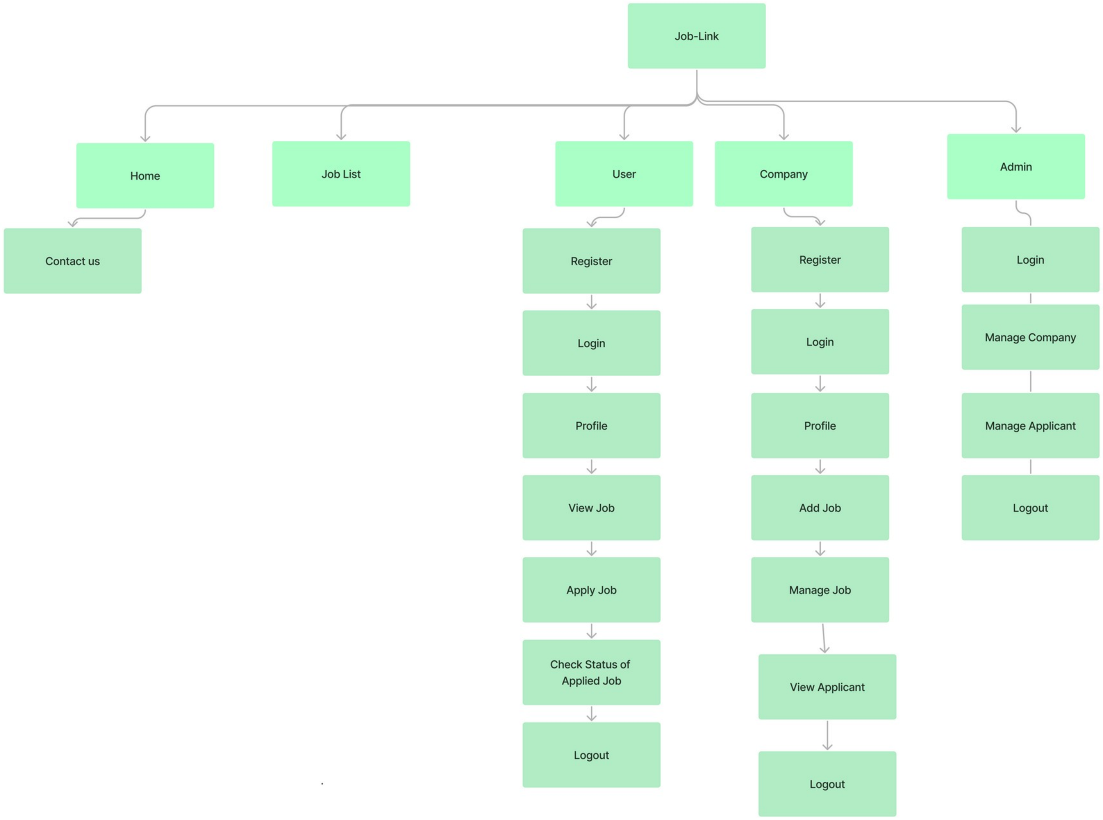
4.4 Class Diagram



4.5 Deployment diagram

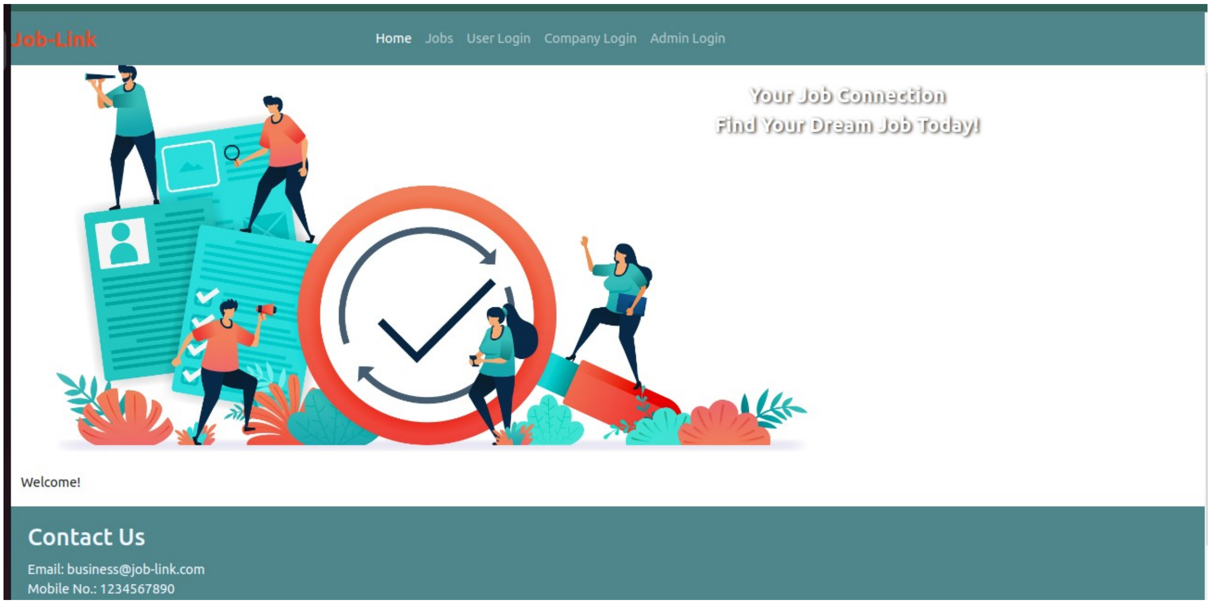


4.6 Website Map Diagram



4.7 User Interface Design (Screens)

//Home Page

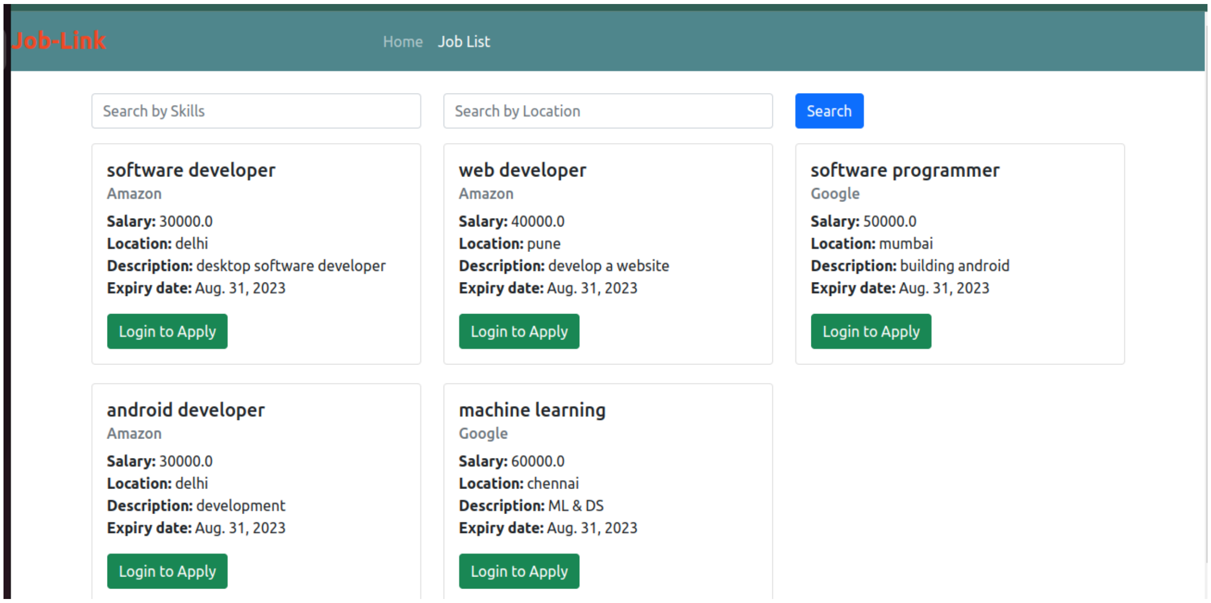


Welcome!

Contact Us

Email: business@job-link.com
Mobile No.: 1234567890

//Job List



software developer
Amazon
Salary: 30000.0
Location: delhi
Description: desktop software developer
Expiry date: Aug. 31, 2023

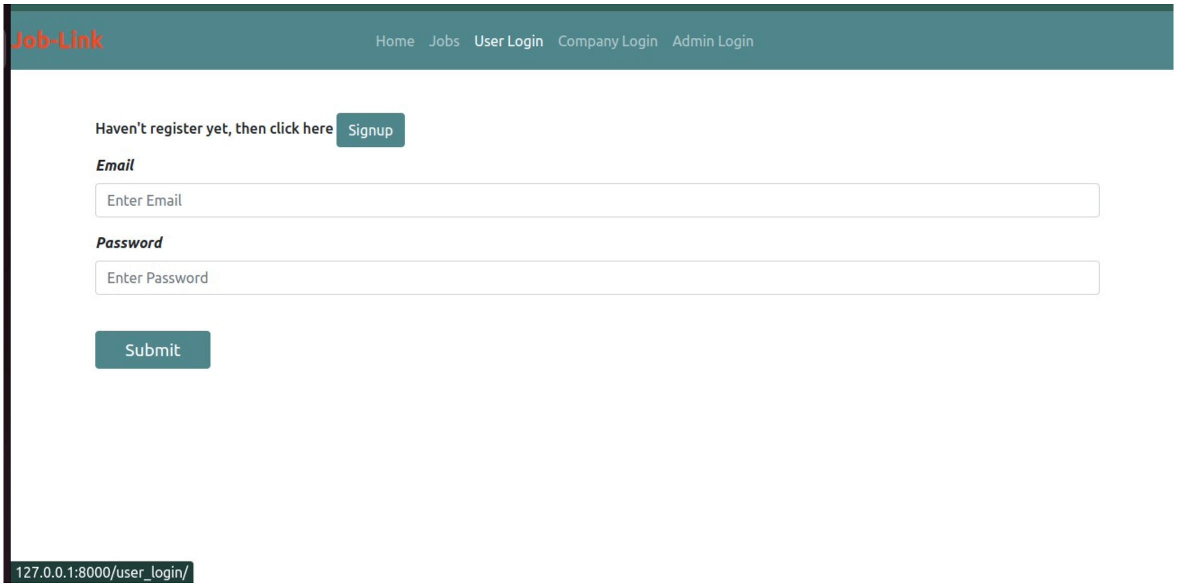
web developer
Amazon
Salary: 40000.0
Location: pune
Description: develop a website
Expiry date: Aug. 31, 2023

software programmer
Google
Salary: 50000.0
Location: mumbai
Description: building android
Expiry date: Aug. 31, 2023

android developer
Amazon
Salary: 30000.0
Location: delhi
Description: development
Expiry date: Aug. 31, 2023

machine learning
Google
Salary: 60000.0
Location: chennai
Description: ML & DS
Expiry date: Aug. 31, 2023

//Login Page



Haven't register yet, then click here Signup

Email

Enter Email

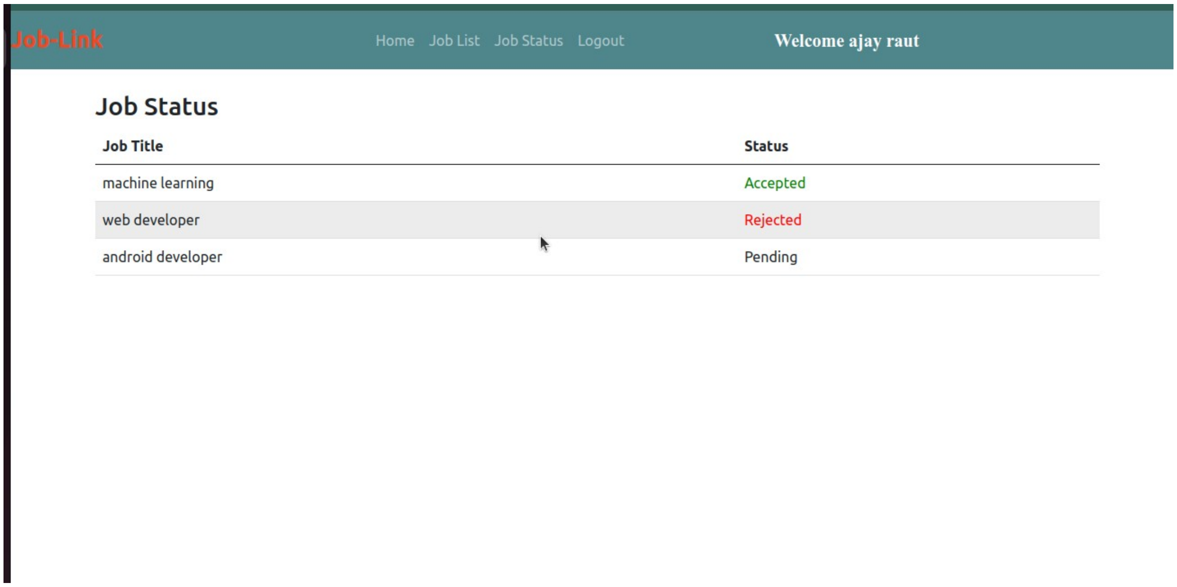
Password

Enter Password

Submit

127.0.0.1:8000/user_login/

//User/Applicant Page

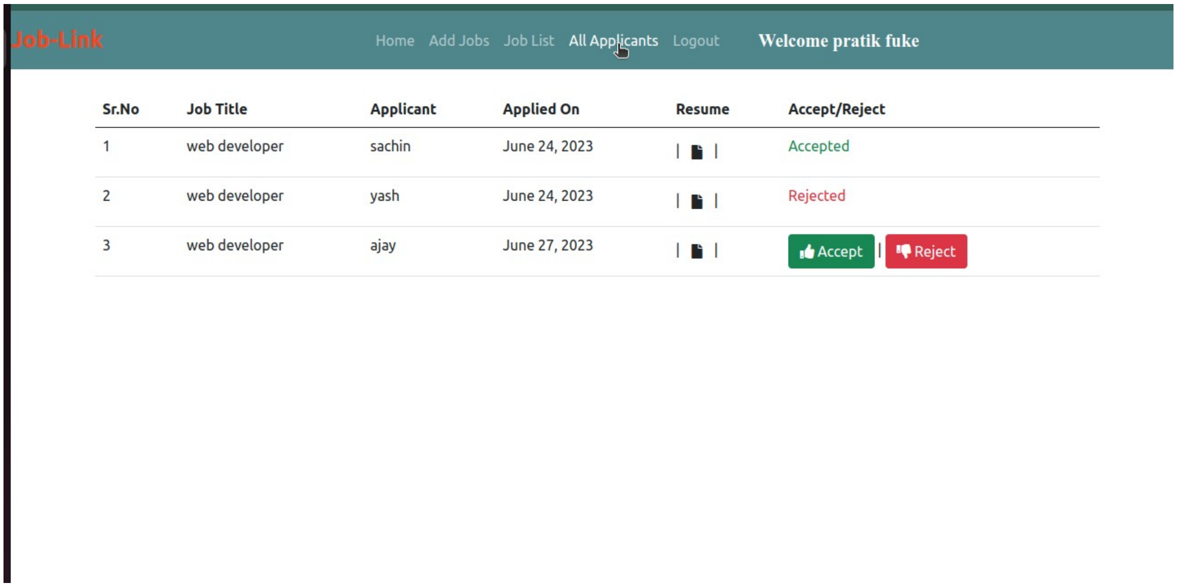


Welcome ajay raut

Job Status

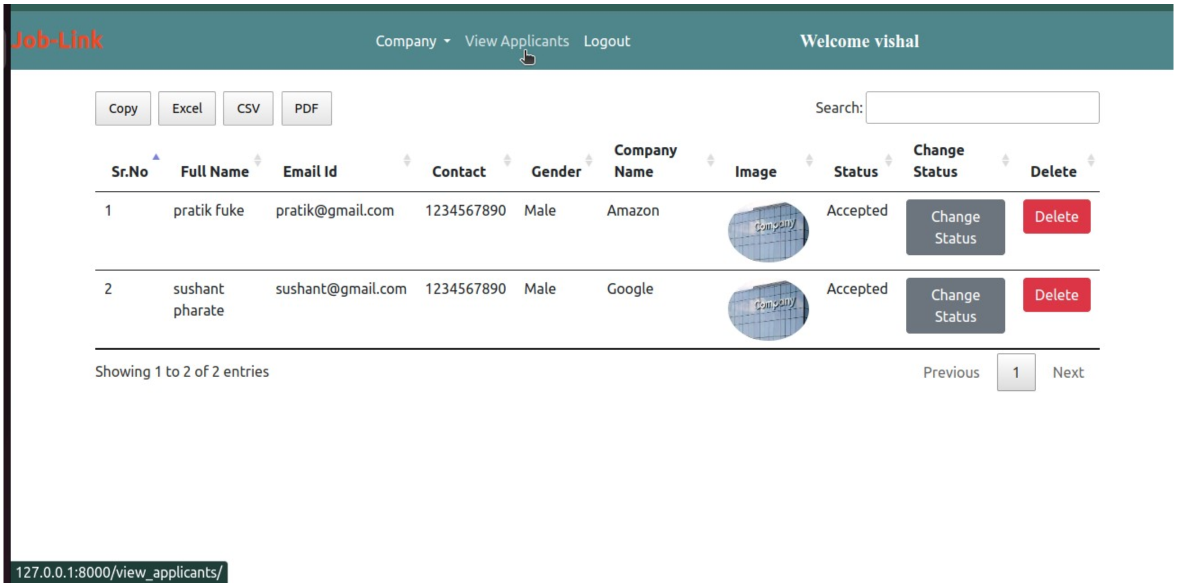
Job Title	Status
machine learning	Accepted
web developer	Rejected
android developer	Pending

//Company page



Sr.No	Job Title	Applicant	Applied On	Resume	Accept/Reject
1	web developer	sachin	June 24, 2023	| 📄 |	Accepted
2	web developer	yash	June 24, 2023	| 📄 |	Rejected
3	web developer	ajay	June 27, 2023	| 📄 |	👍 Accept | 👎 Reject

//Admin page

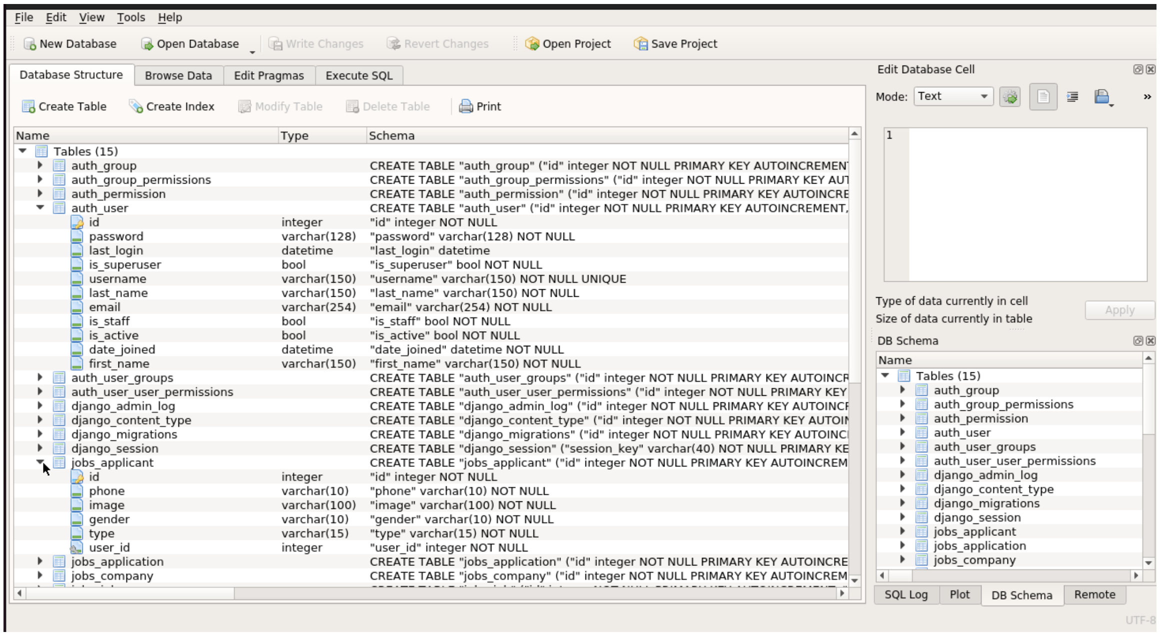


Sr.No	Full Name	Email Id	Contact	Gender	Company Name	Image	Status	Change Status	Delete
1	pratik fuke	pratik@gmail.com	1234567890	Male	Amazon		Accepted	Change Status	Delete
2	sushant pharate	sushant@gmail.com	1234567890	Male	Google		Accepted	Change Status	Delete

Showing 1 to 2 of 2 entries

4.8 Database tables and structure

//Database(SQLite3)



Name	Type	Schema
Tables (15)		
auth_group		CREATE TABLE "auth_group" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT
auth_group_permissions		CREATE TABLE "auth_group_permissions" ("id" integer NOT NULL PRIMARY KEY AUT
auth_permission		CREATE TABLE "auth_permission" ("id" integer NOT NULL PRIMARY KEY AUTOINCRE
auth_user		CREATE TABLE "auth_user" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
id	integer	"id" integer NOT NULL
password	varchar(128)	"password" varchar(128) NOT NULL
last_login	datetime	"last_login" datetime
is_superuser	bool	"is_superuser" bool NOT NULL
username	varchar(150)	"username" varchar(150) NOT NULL UNIQUE
last_name	varchar(150)	"last_name" varchar(150) NOT NULL
email	varchar(254)	"email" varchar(254) NOT NULL
is_staff	bool	"is_staff" bool NOT NULL
is_active	bool	"is_active" bool NOT NULL
date_joined	datetime	"date_joined" datetime NOT NULL
first_name	varchar(150)	"first_name" varchar(150) NOT NULL
auth_user_groups		CREATE TABLE "auth_user_groups" ("id" integer NOT NULL PRIMARY KEY AUTOINCR
auth_user_user_permissions		CREATE TABLE "auth_user_user_permissions" ("id" integer NOT NULL PRIMARY KEY
django_admin_log		CREATE TABLE "django_admin_log" ("id" integer NOT NULL PRIMARY KEY AUTOINCR
django_content_type		CREATE TABLE "django_content_type" ("id" integer NOT NULL PRIMARY KEY AUTOIN
django_migrations		CREATE TABLE "django_migrations" ("id" integer NOT NULL PRIMARY KEY AUTOINC
django_session		CREATE TABLE "django_session" ("session_key" varchar(40) NOT NULL PRIMARY KE
jobs_applicant		CREATE TABLE "jobs_applicant" ("id" integer NOT NULL PRIMARY KEY AUTOINCREM
id	integer	"id" integer NOT NULL
phone	varchar(10)	"phone" varchar(10) NOT NULL
image	varchar(100)	"image" varchar(100) NOT NULL
gender	varchar(10)	"gender" varchar(10) NOT NULL
type	varchar(15)	"type" varchar(15) NOT NULL
user_id	integer	"user_id" integer NOT NULL
jobs_application		CREATE TABLE "jobs_application" ("id" integer NOT NULL PRIMARY KEY AUTOINCRE
jobs_company		CREATE TABLE "jobs_company" ("id" integer NOT NULL PRIMARY KEY AUTOINCREM

Name
Tables (15)
auth_group
auth_group_permissions
auth_permission
auth_user
auth_user_groups
auth_user_user_permissions
django_admin_log
django_content_type
django_migrations
django_session
jobs_applicant
jobs_application
jobs_company

//User table

Name	**Type**	**Key**	**Schema**	**Description**
user_id	int	Primary key auto-increment	Not Null	Applicant User ID
password	varchar(128)	-	Not Null	User password
username	varchar(150)	Unique key	Not Null	Username for Applicant
email	varchar(150)	-	Not Null	Email id of user
first_name	varchar(150)	-	Not Null	First name of user
last_name	varchar(254)	-	Not Null	Last name of user
is_superuser	bool	-	Not Null	User Is admin or not
last_login	datetime	-	Null	Last login date of user

//Company table

Name	**Type**	**Key**	**Schema**	**Description**
company_id	int	Primary key auto-increment	Not Null	Company ID
phone	varchar(10)	-	Not Null	Phone no of company user
image	varchar(100)	-	Not Null	Logo of company
gender	varchar(15)	-	Not Null	Gender of company user
type	varchar(20)	-	Not Null	User type(applicant/Company)
status	varchar(100)	-	Not Null	Company acceptance status
company_name	int	Foreign key (user_id)	Not Null	User and company relation

//Job table

Name	**Type**	**Key**	**Schema**	**Description**
job_id	int	Primary key auto-increment	Not Null	Job id
start_date	date	-	Not Null	Start date of job post
end_date	date	-	Not Null	End date of job post
tittle	varchar(200)	-	Not Null	Tittle of job
salary	real	-	Not Null	Salary for job
image	varchar(100)	-	Not Null	Job company logo
description	text	-	Not Null	Description about job
experience	varchar(100)	-	Not Null	Experience required for job
location	varchar(100)	-	Not Null	Location of job
skill	varchar(200)	-	Not Null	Skill required for job
company_id	int	Foreign key (company_id)	Not Null	Company and job relation

5. Coding

During the coding phase of the Job-Link online job portal project, a comprehensive implementation is carried out to bring the system to life. This phase involves writing code using various technologies, languages, and frameworks. Here's an expanded and more detailed description:

Front-End Development:

- HTML: HTML (Hypertext Markup Language) is used to create the structure and content of the web pages. It defines the elements on the pages, such as headings, paragraphs, lists, forms, buttons, and images. HTML provides the foundation for the user interface and the layout of the Job-Link portal.

- CSS: CSS (Cascading Style Sheets) is employed to style and format the appearance of the web pages. It controls the colors, fonts, sizes, spacing, and other visual aspects of the user interface, ensuring a consistent and visually appealing design throughout the portal. CSS helps in creating responsive and mobile-friendly layouts.

- JavaScript: JavaScript is utilized for client-side interactivity and dynamic functionalities on the web pages. It enables features like form validation, dynamic content updates, real-time interactions, AJAX requests, and DOM manipulation. JavaScript enhances the user experience by adding interactivity and responsiveness to the Job-Link portal.

Back-End Development:

- Python: Python serves as the primary programming language for the back-end development of the Job-Link portal. It is a versatile language known for its readability and ease of use. Python is used to handle the server-side logic, process data, and interact with various components of the system. It enables the implementation of business rules, algorithms, and complex computations required by the application.

- Django: Django, a powerful Python web framework, is used to streamline the development process and provide a robust foundation for the back-end of the Job-Link portal. It follows the Model-View-Controller (MVC) architectural pattern, emphasizing code reusability and a clean design. Django handles URL routing, request handling, session management, authentication, and database interactions, making it easier to build scalable and maintainable web applications.

- Django Templates: Django templates are used to create dynamic HTML pages by embedding Python code within HTML templates. This enables the generation of dynamic content based on data retrieved from the server. Django templates support loops, conditionals, and template inheritance, allowing for efficient and flexible rendering of HTML pages.

- Django Views: Django views are responsible for handling requests from the front-end and returning appropriate responses. They encapsulate the business logic of the application, processing data, interacting with models, and generating the appropriate output. Views retrieve data from the database, perform calculations, and orchestrate the flow of information between the front-end and back-end components.

- Django Models: Django models define the data models and database schemas using Python classes. They represent the entities and relationships within the Job-Link system, such as Job Seekers, Employers, Job Listings, and Applications. Django's Object-Relational Mapping (ORM) allows developers to work with the database using Python code, abstracting the underlying SQL queries and simplifying database interactions.

Database:

- SQLite: SQLite is used as the database management system during the development and testing phases of the Job-Link project. It is a lightweight, file-based relational database that integrates seamlessly with Django. SQLite provides an efficient and self-contained solution for storing and retrieving data.

Throughout the coding phase, we follow coding standards, best practices, and adhere to the project's design specifications. we implement the required features, functionalities, and interactions specified in the project requirements.

The front-end code is responsible for creating the visual presentation and user experience, while the back-end code handles data processing, authentication, business rules, and database interactions.

The Django framework facilitates the integration of these components and ensures the seamless functioning of the Job-Link portal.

As the project progresses, additional libraries, frameworks, or technologies might be incorporated based on the project's needs.

The coding phase is crucial as it transforms the design and specifications into a functional system. Thorough testing and quality assurance processes are carried out to identify and resolve any bugs or issues in the code before moving on to the testing and deployment stages.

6. Testing

The testing phase in the Job-Link project is crucial to ensure that the system functions as intended, meets the requirements, and provides a reliable and user-friendly experience. Here are the key aspects of the testing phase:

- Test Planning: Define the testing objectives, scope, and test strategy. Identify the test scenarios, data, and resources needed for testing. Create a test plan that outlines the testing approach and timelines.

- Test Case Development: Develop test cases based on the system requirements, including functional and non-functional aspects. Test cases should cover various scenarios, inputs, and expected outputs to validate the system's behavior.

- System Testing: Conduct end-to-end testing of the entire system to validate its functionality, usability, performance, security, and compatibility. Test various user scenarios, user interfaces, and system interactions.

- User Acceptance Testing (UAT): Involve actual users, including job seekers and employers, to test the system in a real-world environment. Gather feedback, validate user requirements, and ensure the system meets user expectations.

- Unit Testing: Test individual components of the system, such as functions, modules, and classes, to ensure they work correctly in isolation. Use techniques like white-box testing and code reviews to verify the code quality.

- **Testing Table**

Test Case	**Data Input**	**Expected Output**	**Actual Output**	**Pass/Fail**
1	Valid credentials	Successful login	Successful login	pass
2	Invalid credentials	Error message displayed	Error message displayed	pass
3	Empty ‘username’ field	Error message displayed	Error message displayed	pass
4	Empty ‘password’ field	Error message displayed	Error message displayed	pass
5	Valid job search keyword	Relevant job listings displayed	Relevant job listings displayed	pass
6	Invalid job search keyword	No job listings displayed	No job listings displayed	pass
7	Apply for a job	Confirmation message displayed	Confirmation message displayed	pass
8	Access restricted admin feature as regular user	Error message displayed	Error message displayed	pass

7. Limitations and Enhancements

7.1 Limitations of the Job-Link system:

1. Company Verification:

   The Job-Link system does not have a built-in feature for company verification through the submission and review of legal documents. As a result, the system cannot guarantee the authenticity and legitimacy of the companies listed on the platform, potentially allowing for the inclusion of fraudulent or misrepresented entities.

2. Connecting Interface:

   The Job-Link system does not provide a dedicated connecting interface between users/applicants and companies for future job scheduling and interviews. This limitation hinders direct and streamlined communication between the parties, making it more challenging to coordinate job-related activities efficiently.

3. Job Recommendation:

   The Job-Link system does not incorporate a job recommendation system based on algorithms or historical data. As a result, it does not provide personalized or up-to-date job suggestions to users/applicants based on their profiles or preferences.

4. Social Media Integration:

   The Job-Link system does not integrate social media features or leverage social media profiles for applicant information. It lacks the ability to connect with social media platforms or access relevant applicant data from these sources, limiting the comprehensiveness and insights provided by applicant profiles.

7.2 Future Enhancements to the Job-Link system:

1. Advanced Search and Filtering: Implementing advanced search and filtering options based on job categories, locations, salary ranges, and other criteria would enhance the user experience and help job seekers find relevant opportunities more efficiently.
2. Company Verification:

   Strengthen the company verification process to ensure legitimate and authentic companies are listed on the platform.
3. Connecting Interface:

   Improve the user interface for seamless communication between users and companies, making job scheduling and interview coordination easier.
4. Job Recommendation:

   Implement an advanced job recommendation system that suggests personalized and up-to-date job opportunities based on user preferences and market trends.
5. Social Media Integration:

   Integrate social media platforms to enhance applicant profiles, showcasing their professional information and achievements.

8. Conclusion

In conclusion, the Job-Link project aims to develop an online job portal that connects job seekers with employers, facilitating the job search and recruitment process. The proposed system includes features such as job search, application submission, employer profiles, and user profiles.

The project's objectives focus on providing a user-friendly interface, efficient job matching, and effective communication between job seekers and employers. The system aims to streamline the job search process, increase accessibility to job opportunities, and enhance the overall user experience.

During the testing phase, various techniques such as functional testing, integration testing, performance testing, and security testing are employed to ensure the system's reliability, functionality, and security. User acceptance testing is conducted to gather feedback and validate the system's compliance with user requirements.

While the system has limitations, such as limited job market coverage and potential language constraints, enhancements can be made to expand job market coverage, improve employer engagement, and provide multilingual and multicultural support.

Overall, the Job-Link project offers a promising solution to facilitate the job search and recruitment process, bringing job seekers and employers together in a user-friendly and efficient online platform. With continuous improvement and enhancements, the system has the potential to make a positive impact in the field of job placement and career development.

9. Bibliography

- Google

  https://www.google.com

- YouTube

  https://www.youtube.com

- Python

  https://docs.python.org

- Django project

  https://www.djangoproject.com/

- SQLlite

  https://www.sqlite.org/index.html

- Stack overflow

  https://stackoverflow.com

- w3schools

  https://www.w3schools.com

10. Annexure

- Sample Code
  models.py

```
from django.db import models
from django.contrib.auth.models import User

class Applicant(models.Model):
user = models.ForeignKey(User, on_delete=models.CASCADE)
phone = models.CharField(max_length=10)
image = models.ImageField(upload_to="")
gender = models.CharField(max_length=10)
type = models.CharField(max_length=15)

def __str__(self):
return self.user.first_name

class Company(models.Model):
user = models.ForeignKey(User, on_delete=models.CASCADE)
phone = models.CharField(max_length=10)
image = models.ImageField(upload_to="")
# company_doc = models.ImageField(upload_to="", default="default_image.jpg")
gender = models.CharField(max_length=10)
type = models.CharField(max_length=15)
status = models.CharField(max_length=20)
company_name = models.CharField(max_length=100)

def __str__(self):
return self.user.username

class Job(models.Model):
company = models.ForeignKey(Company, on_delete=models.CASCADE)
start_date = models.DateField()
end_date = models.DateField()
title = models.CharField(max_length=200)
salary = models.FloatField()
```

```
image = models.ImageField(upload_to="")
description = models.TextField(max_length=400)
experience = models.CharField(max_length=100)
location = models.CharField(max_length=100)
skills = models.CharField(max_length=200)
creation_date = models.DateField()

def __str__(self):
return self.title

class Application(models.Model):
company = models.CharField(max_length=200, default="")
job = models.ForeignKey(Job, on_delete=models.CASCADE)
applicant = models.ForeignKey(Applicant, on_delete=models.CASCADE)
resume = models.ImageField(upload_to="")
apply_date = models.DateField()
status = models.CharField(max_length=20, default="Pending")

def __str__(self):
return str(self.applicant)
```