JOB HUNT:
Extracting opportunities from web

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

CSE (DATA SCIENCE)

By

JULURU CHAITANYA SAI (22AG1A6795)
B .KOULINYA SHARVANI (22AG1A67A3)
CHIPPABATHINI VIKAS (22AG1A6780)

Under the guidance of
P.NIHARIKA

DEPARTMENT OF CSE (DATA SCIENCE)

ACE Engineering College
Ankushapur(V), Ghatkesar(M), Medchal Dist - 501 301
(An Autonomous Institution, Affiliated to JNTUH, Hyderabad)
www.aceec.ac.in
2023-2024

ACE

Engineering College

UGC AUTONOMOUS INSTITUTION

(Sponsored by Yadala Satyanarayana Memorial Educational Society, Hyderabad)

Approved by AICTE & Affiliated to JNTUH

B.Tech Courses offered: CIVIL, CSE, IT, ECE, EEE & MECH, NBA Accredited Courses: CIVIL, CSE, ECE, EEE & MECH, Accorded NAAC A - Grade

DEPARTMENT OF CSE (DATA SCIENCE)

CERTIFICATE

This is to certify that the Societal Related (Or) Real Time project report entitled “project title” is a bonafide work done by Jurulu Chaitanya Sai , B.Koulinya Sharvani and Chippabathini Vikas bearing 22AG1A6795, 22AG1A67A3, 22AG1A6780 in partial fulfillment for the award of Degree of BACHELOR OF TECHNOLOGY in CSE (Data Science) from JNTUH University, Hyderabad during the academic year 2023- 2024. This record of bonafide work carried out by them under our guidance and supervision.

The results embodied in this report have not been submitted by the student to any other University or Institution for the award of any degree or diploma.

P.NIHARIKA	**(Dr. P. Chiranjeevi)**
Professor	Associate Professor

Supervisor H.O.D., CSE-DS.

ACKNOWLEDGEMENT

We would like to express my gratitude to all the people behind the screen who have helped me transform an idea into a real time application.

We would like to express my heart-felt gratitude to my parents without whom we would not have been privileged to achieve and fulfill my dreams.

A special thanks to our General Secretary, **Prof. Y. V. Gopala Krishna Murthy,** for having founded such an esteemed institution. Sincere thanks to our Joint Secretary **Mrs. M. Padmavathi**, for support in doing project work. I am also grateful to our beloved principal, **Dr. B. L. RAJU** for permitting us to carry out this project.

We profoundly thank **Dr. P. Chiranjeevi**, Associate Professor and Head of the Department of Computer Science and Engineering (Data Science), who has been an excellent guide and also a great source of inspiration to my work.

We extremely thank **Mrs. B. Saritha** and **Mr. M. Hari Krishna,** Assistant Professors, Project coordinators, who helped us in all the way in fulfilling of all aspects in completion of our Mini-Project.

We are very thankful to my internal guide (P. Niharika) who has been an excellent and also given continuous support for the Completion of my project work.

The satisfaction and euphoria that accompany the successful completion of the task would be great, but incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success. In this context, I would like to thank all the other staff members, both teaching and non-teaching, who have extended their timely help and eased my task.

JULURU CHAITANYA SAI (22AG1A6795)
B .KOULINYA SHARVANI (22AG1A67A3)
CHIPPABATHINI VIKAS (22AG1A6780)

JOB HUNT:
Extracting opportunities from web

ABSTRACT

The increasing demand for job opportunities there is the need for efficient data retrieval from online job portals such as LinkedIn and Indeed.

This project focuses on Designing and developing a web application which web scrapes the jobs in different web portals and notifies the user about job applications based on user's choice.

Web scraping application plays a major role in this project which is capable of systematically retrieving job postings from LinkedIn/Indeed. It is achieved by using Python libraries such as BeautifulSoup , Scrapy and PHP to get the data, which is backend. HTML and CSS is used for structuring the website and extract relevant information such as job titles, descriptions, locations, and application deadlines. The scraping tool successfully retrieves job listings from the Job Portal and organizing them into a structured format for the users. The extracted data includes essential details required by job seekers and recruiters, providing a overview of available opportunities.

Synopsis:

By automating the process of gathering job listings from Web portals , scraping tool efficiently does the search process for users and facilitates data-driven decision-making for recruiters and job seekers. The project underscores the significance of web scraping in accessing valuable information from online sources, providing the way for enhanced efficiency and effectiveness in the job market.

CONTENT

CONTENT

S.NO	CHAPTER NAME	PAGE NO
1	INTRODUCTION	1
2	EXISTING SYSTEM & DRAWBACKS	3
3	LITERATURE SURVEY	5
4	PROPOSED MODEL	8
5	REQUIREMENTS GATHERING	10
5.1	Software Requirements & Hardware Requirements	10
5.2	Functional Requirements & Non-Functional Requirements	11
5.3	Data Collection	12
6	SYSTEM ANALYSIS & DESIGN	14
6.1	Module Description	14
6.2	System Architecture	16
6.3	UML Diagrams	19

7 IMPLEMENTATION 25

7.1 Algorithm Used 25

7.2 Sample Code 26

8 TESTING 28

9 RESULTS 32

10 CONCLUSION 40

FUTURE WORK 41

REFERENCES 42

LIST OF ABBREVIATIONS

Abbreviation	**Full Form**
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
PHP	Hypertext Preprocessor
API	Application Programming Interface
AI	Artificial intelligence
IDE	Integrated development environment
CSV	Comma-separated values
ERD	Entity relationship diagram
URL	Uniform Resource Locator
UML	Unified Modeling Language

1.INTRODUCTION

The Growing Demand for Job Opportunities

In today's competitive job market, the demand for job opportunities is continually increasing. Job seekers are constantly on the lookout for suitable positions that match their skills and preferences. However, manually searching through numerous job portals can be time-consuming and overwhelming.

The Need for Efficient Data Retrieval

To address this challenge, there is a need for efficient data retrieval systems that can systematically gather job postings from various online job portals such as LinkedIn and Indeed. By automating this process, can save time and effort for job seekers and provide them with up-to-date information about available job opportunities.

Project Overview

This project focuses on designing and developing a web application capable of web scraping jobs from different web portals and notifying users about job applications based on their preferences. The web scraping application plays a crucial role in this project, systematically retrieving job postings and organizing them into a structured format.

Technologies and Tools

To achieve this, plan to use Python libraries such as BeautifulSoup ,Scrapy and PHP for data extraction. These libraries will handle the backend operations of retrieving job listings from LinkedIn and Indeed. HTML and CSS will be used for structuring the website and extracting relevant information such as job titles, descriptions, locations, and application deadlines.

Functionality and Features

The scraping tool will successfully retrieve job listings from the job portals and organize them into a structured format for users. The extracted data will include essential details required by job seekers and recruiters, providing a comprehensive overview of available opportunities

.

2. EXISTING SYSTEM

The current landscape of job search portals like LinkedIn and Indeed is marked by a burgeoning demand for efficient data retrieval solutions. Addressing this need, this project focuses on the design and development of a web application aimed at systematically scraping job postings from these platforms and notifying users about relevant job opportunities based on their preferences.

Utilizing Python libraries such as BeautifulSoup, Scrapy and PHP the backend architecture is designed to efficiently gather data by extracting key details like job titles, descriptions, locations, and application deadlines. This data is then structured using HTML and CSS to create a user-friendly interface that enhances the job search experience.

The scraping tool acts as a robust mechanism for retrieving and organizing job listings, catering to the diverse needs of both job seekers and recruiters. By automating the search process, this project not only streamlines job discovery but also supports data-driven decision-making in the job market, emphasizing the pivotal role of web scraping in accessing and utilizing valuable information from online sources.

Existing system drawbacks:

Reliability of Data

Job listings on these portals can change frequently, leading to issues with data freshness and accuracy. Scraped data may quickly become outdated if not updated regularly.

Complexity of Maintenance

Maintaining a web scraping system requires continuous monitoring and adjustments. Changes in the structure of the job portal websites can break the scraping scripts, requiring updates to the scraping logic.

Time Complexity

Depending on the size of job listings and the frequency of updates, the time complexity of scraping can vary. Algorithms and data structures used in scraping scripts (like using efficient data structures for storage and retrieval) play a crucial role in managing this complexity.

User Experience Challenges:

Depending solely on scraped data might not provide the best user experience compared to using official APIs provided by job portals, which are more reliable and structured.

Scalability:

Scaling up the scraping process to handle a large number of users concurrently can be challenging without robust infrastructure and optimization techniques.

3. LITERATURE SURVEY

The literature survey gives the key technologies and methodologies involved in designing and developing a web application for job scraping and user notifications. Utilizing Python libraries like BeautifulSoup, Scrapy and PHP for data extraction, HTML and CSS for web page structuring, and backend frameworks like Flask or Django for integration, the project aims to provide an efficient solution for job seekers to stay informed about relevant job opportunities.

A literature survey on job hunting through web extraction examines the strategies and technologies utilized to gather job opportunities from online sources. It encompasses research on web scraping algorithms. Scholars investigate the effectiveness of various extraction methods, the accuracy of job information retrieved, and the usability of extracted data for job seekers and recruiters alike. These surveys often identify trends, challenges, and advancements in the field, offering valuable insights for improving job search platforms and services.

Despite the technological advancements, several challenges and ethical considerations need to be addressed. Technical challenges include ensuring data quality, scalability of the scraping and processing systems, and maintaining up-to-date information. Ethical considerations revolve around data privacy and compliance with regulations such as GDPR. The ethical use of data and fairness in job recommendation systems are also critical issues that need careful consideration to avoid biases and ensure equitable opportunities for all job seekers.

Reviewing existing systems and platforms, such as Indeed and LinkedIn, provides valuable insights into the practical applications of these technologies. These platforms have successfully implemented various aspects of web scraping to create efficient job search systems. Case studies of these platforms reveal best practices, common pitfalls, and innovative approaches that can inform future research and development.

In conclusion, the automation of job searches through web scraping, presents significant opportunities to enhance the job search process. However, it also poses several technical and ethical challenges that need to be carefully managed. Future research should focus on improving the accuracy and fairness of job matching algorithms, addressing data privacy concerns, and developing scalable solutions that can adapt to the dynamic nature of online job postings. By continuing to innovate and address these challenges, we can create more effective and equitable job search systems that benefit both job seekers and employers.

CITATIONS-

The paper by **Palanivel, K., et al. (2016) titled "Web Scraping using Python"** published in the *International Journal of Computer Applications* discusses the fundamentals and applications of web scraping using Python.The authors highlight Python's effectiveness as a tool for web scraping due to its simplicity and the availability of powerful libraries like Beautiful Soup and Scrapy. The paper also explores different techniques for extracting information, handling dynamic content, and managing challenges like anti-scraping measures and legal considerations.

The paper by **Zhang (2020), titled "Large-scale Job Data Extraction Using Scrapy: A Case Study on Indeed,"** published in the *International Journal of Web Information Systems,* discusses the implementation of a large-scale job data extraction system using the Scrapy framework. The study focuses on how Scrapy can be effectively utilized to scrape job listings from Indeed, addressing challenges like handling dynamic content, managing IP bans, and optimizing the scraping process for efficiency. The paper provides insights into the architecture of the scraping system, data preprocessing techniques, and the application of machine learning models to classify and organize the extracted job data for further analysis.

The article **"Job Market Analysis Using Web Scraping: A Study on LinkedIn" by S. Gupta (2021)** explores the application of web scraping techniques to analyze job market trends using data extracted from LinkedIn. The study employs Python-based tools to collect and process job listing data, offering insights into demand for specific skills, job roles, and industry trends. By analyzing this data, the paper highlights the potential of web scraping in providing valuable information for job seekers and employers, contributing to more informed decision-making in the job market. The findings emphasize the growing importance of data-driven approaches in understanding labor market dynamics.

4. PROPOSED MODEL

Advanced Filtering and Search Options:

Enhance the search functionality with advanced filters such as remote work options, flexible hours, company culture, diversity policies, and more. Allow users to prioritize what matters most to them.

Benefits:

- Provides a more tailored job search experience.
- Helps users find job opportunities that align with their personal and

Resume and Portfolio Integration:

Allow users to create and upload resumes and digital portfolios showcasing their skills, projects, and achievements. Incorporate tools for creating high-quality video content.

Benefits:

- Helps job seekers stand out with a unique and personal touch.
- Provides employers with a more comprehensive view of a candidate's capabilities.

Company Insights and Reviews:

Offer detailed company profiles with insights into company culture, employee reviews, benefits, and career growth opportunities.

Benefits:

- Helps users make informed decisions about potential employers.
- Provides transparency about workplace conditions and company reputation.

Interactive Interview Practice:

Provide an interactive platform for practicing interview skills. Use AI to simulate interview scenarios, provide feedback, and offer tips for improvement.

Benefits:

- Prepares job seekers for real interview situations.
- Increases confidence and improves interview performance.

5. REQUIREMENTS GATHERING

5.1 Software and hardware requirements

Software requirements:

- Front-end languages - HTML, CSS, JavaScript
- Back-end languages -python libraries (Beautiful soup , Scrapy), MySql for data bases and PHP.
- Text editor or IDE for writing code and editing**.**

Hardware requirements:

- Operating System : Windows 11
- System Processor : Intel i3 configuration with storage of 50GB
- RAM: 4GB
- Reliable internet connection

5.2 Functional and Non-Functional requirements

Functional Requirements:

- Ability to search for job listings based on specific criteria (e.g., job title, location, company).
- Extraction of job details such as job title, company name, location, job description, salary, posting date, and application link.
- Handling of dynamic content and multiple pages of listings.
- Data storage in a structured format (e.g., CSV, database).

Non-Functional Requirements:

- Scalability to handle a large number of job listings.
- Accuracy and completeness of the extracted data.
- User-friendly interface for non-technical users.

Technical Feasibility:

- **Tools and Technologies**: Identify the appropriate tools for web scraping (e.g., Beautiful Soup , Scrapy, Selenium) , data storage (e.g., Pandas, databases) and PHP.
- **Infrastructure**: Ensure the availability of sufficient computational resources for running the scraper.

Economic Feasibility:

- **Cost Analysis**: Evaluate the costs associated with development, deployment, and maintenance of the scraping system.
- **Return on Investment**: Consider potential benefits such as improved job search efficiency for users

5.3 Data Collection:

Objectives

- Collect job listings from online job portals such as LinkedIn and Indeed.
- Extract relevant information such as job titles, descriptions, locations, and application deadlines.
- Organize the data in a structured format for user-friendly presentation.

Data Sources

1. LinkedIn: Professional networking site with extensive job listings.
2. Indeed: Comprehensive job search engine with listings from various sources.

Tools and Technologies

- Python Libraries:
- BeautifulSoup: For parsing HTML and XML documents to extract data.
- Scrapy: For building spiders to crawl web pages and retrieve data.
- Web Technologies:
- HTML: For structuring web pages.
- CSS: For styling web pages.

Data Extraction Process

1. **Identifying Target Pages:**

- Determine the URLs of the job listings pages on LinkedIn and Indeed.
- Analyze the structure of these pages to identify the HTML elements containing the desired data.

2. **Building Scrapy Spiders:**
   - Create spiders to crawl the job listings pages and retrieve the HTML content.
   - Configure the spiders to handle pagination and navigate through multiple pages of job listings.
3. **Parsing HTML with BeautifulSoup:**
   - Use BeautifulSoup to parse the retrieved HTML content.
   - Locate and extract specific elements containing job titles, descriptions, locations, and application deadlines.
4. **Data Cleaning and Structuring:**
   - Clean the extracted data to remove unnecessary characters and format it consistently.
   - Organize the data into a structured format, such as a CSV file or a database, for easy access and manipulation**.**

6. SYSTEM ANALYSIS & DESIGN

6.1 Module description:

Module Description: Web Scraping Tool for Job Listings

This module serves as the core backend component of the web application designed to retrieve job listings from online portals such as LinkedIn and Indeed. It leverages PHP and Python libraries like BeautifulSoup and Scrapy for efficient data extraction and structuring.

Key Features:

1. **Data Retrieval & Extraction:** Utilizes web scraping techniques to systematically gather job postings based on user-defined criteria such as job title, location, and keywords. Extracts essential job details including job titles, descriptions, locations, and application deadlines from HTML pages retrieved from job portals.
2. **Data Organization:** Organizes extracted job data into a structured format suitable for presentation to users, ensuring clarity and accessibility.
3. **Automation:** Automates the process of searching and retrieving job listings, reducing manual effort for users and providing timely updates on new job opportunities.
4. **User Notifications:** Facilitates user notifications based on saved job preferences, ensuring users are promptly informed about relevant job openings.
5. **Scalability:** Designed to handle large volumes of data efficiently, scaling as the number of job listings and users grow.

Purpose:

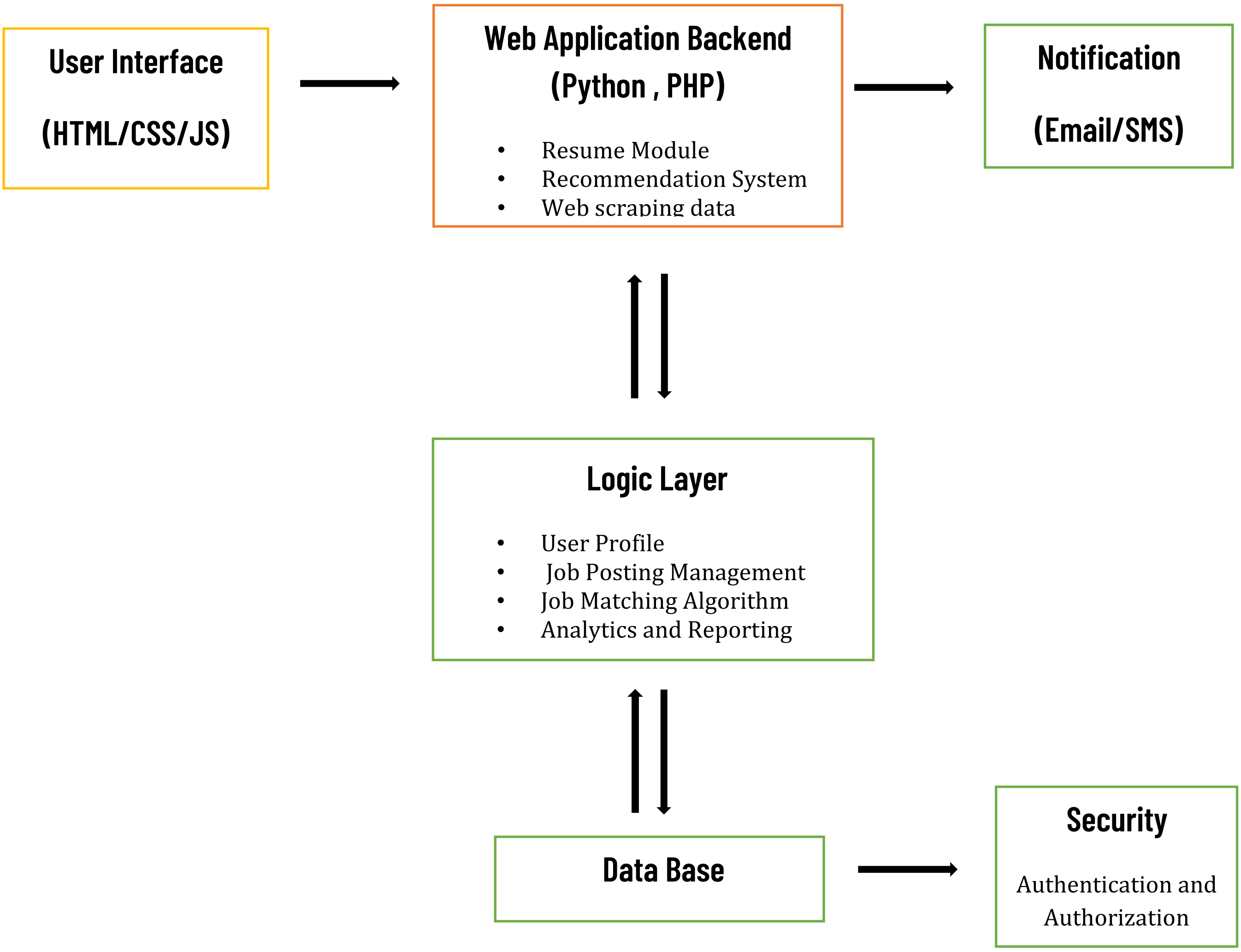
This module plays a crucial role in enhancing efficiency and effectiveness in the job market by providing automated, data-driven insights into available job opportunities. It underscores the significance of web scraping in accessing valuable job market information from online sources.

Implementation: Implemented using Python for backend operations, integrating with HTML and CSS for frontend presentation of job listings and user interface.

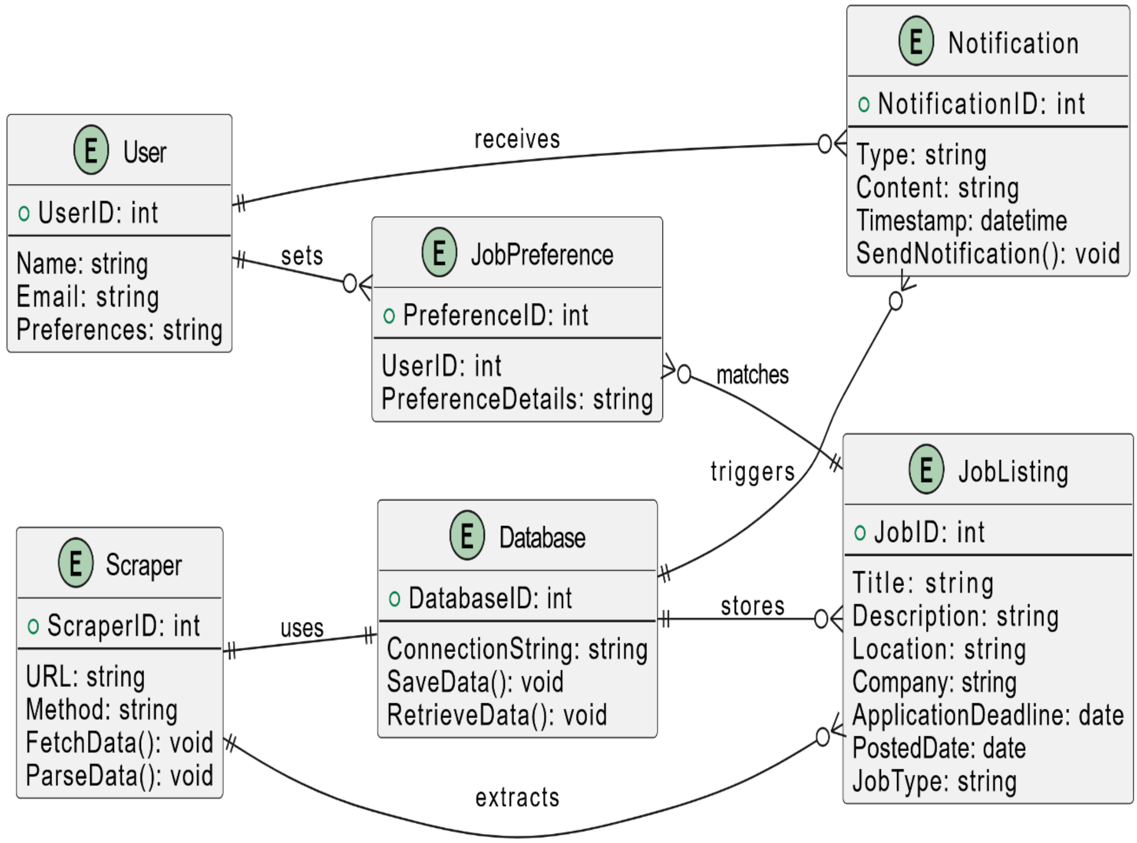
Impact: Enables both job seekers and recruiters to make informed decisions based on up-to-date job market data, promoting streamlined job search processes and recruitment efforts.

Future Enhancements: Potential enhancements include natural language processing for improved job matching, integration with additional job portals, and enhanced user customization features.

6.2 System Architecture: System architecture is the conceptual model that defines the structure, behavior, and more views of a system. It serves as a blueprint for both the system and the project developing it, providing a comprehensive overview of all system components, their relationships, and how they interact to fulfill the system's requirements.



Entity-Relation Diagram: An Entity-Relationship Diagram (ERD) is a visual representation of the data and the relationships between different entities in a database. It is used in database design and helps to model the logical structure of a system.



Explanation of Entities and Relationships

1. User: Represents the users of the application. Each user has a unique ID, name, email, and preferences for job listings.
2. JobListing: Represents job listings retrieved from the job portals. Each job listing has details like title, description, location, company, application deadline, posted date, and job type.
3. Scraper: Represents the web scraping component. It has attributes like URL and method, and methods for fetching and parsing data.
4. Notification: Represents notifications sent to users about new job listings. It includes type, content, timestamp, and a method to send notifications.
5. Database: Represents the storage component for job listings and notifications using PHP. It includes a connection string and methods to save and retrieve data.
6. JobPreference: Represents the preferences set by users for job listings. It links users to their specific preferences.

Relationships

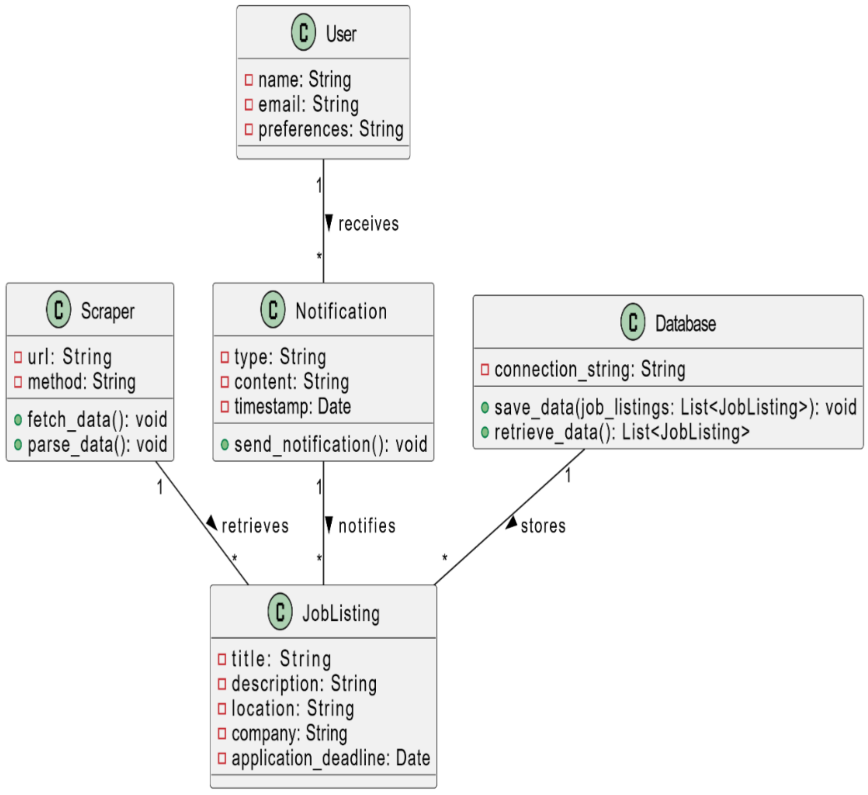
1. User to JobPreference: A user sets multiple job preferences.
2. JobPreference to JobListing: Job preferences match job listings.
3. User to Notification: Users receive multiple notifications.
4. Scraper to Database: The scraper uses the database to store and retrieve data.
5. Scraper to JobListing: The scraper extracts job listings.
6. Database to JobListing: The database stores job listings.
7. Database to Notification: The database triggers notifications.

6.3 UML Diagrams:

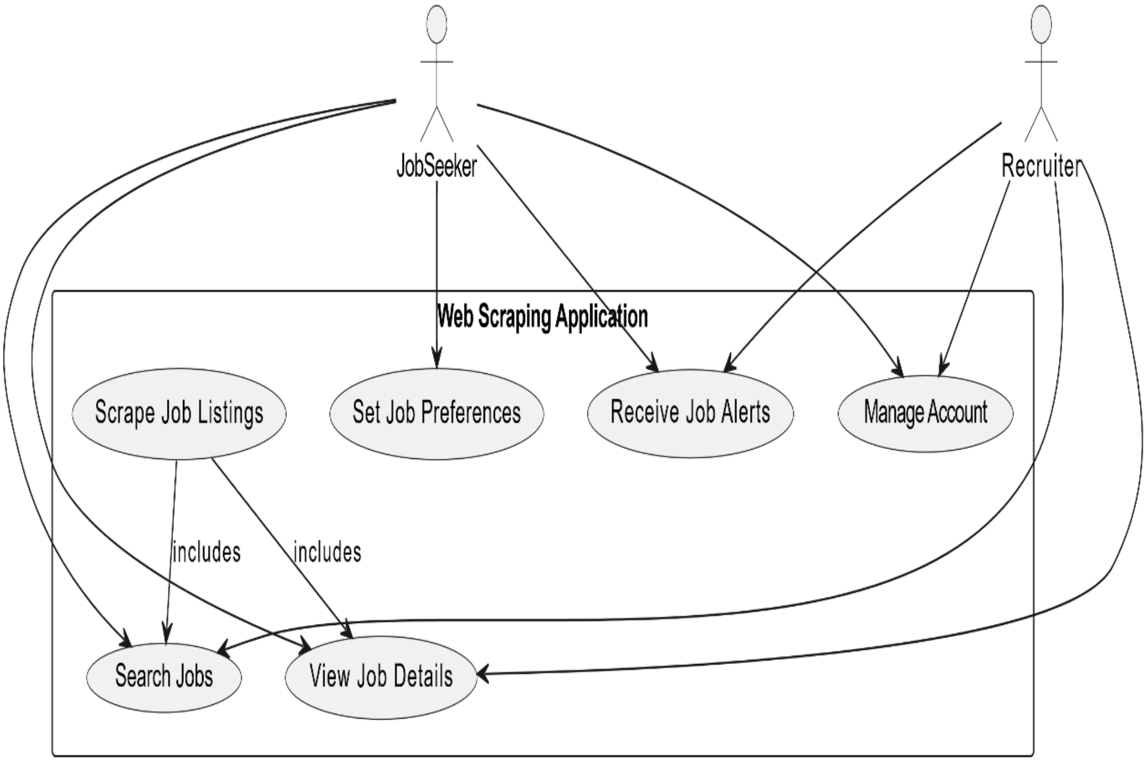
UML:

The Unified Modeling Language (UML) is a standard language for writing software blue prints. The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modeling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modeling yields an understanding of a system.

1. Class Diagram: Class diagrams identify the class structure of a system, including the properties and methods of each class. Also depicted are the various relationships that can exist between classes, such as an inheritance relationship. The Class diagram is one of the most widely used diagrams from the UML specification.

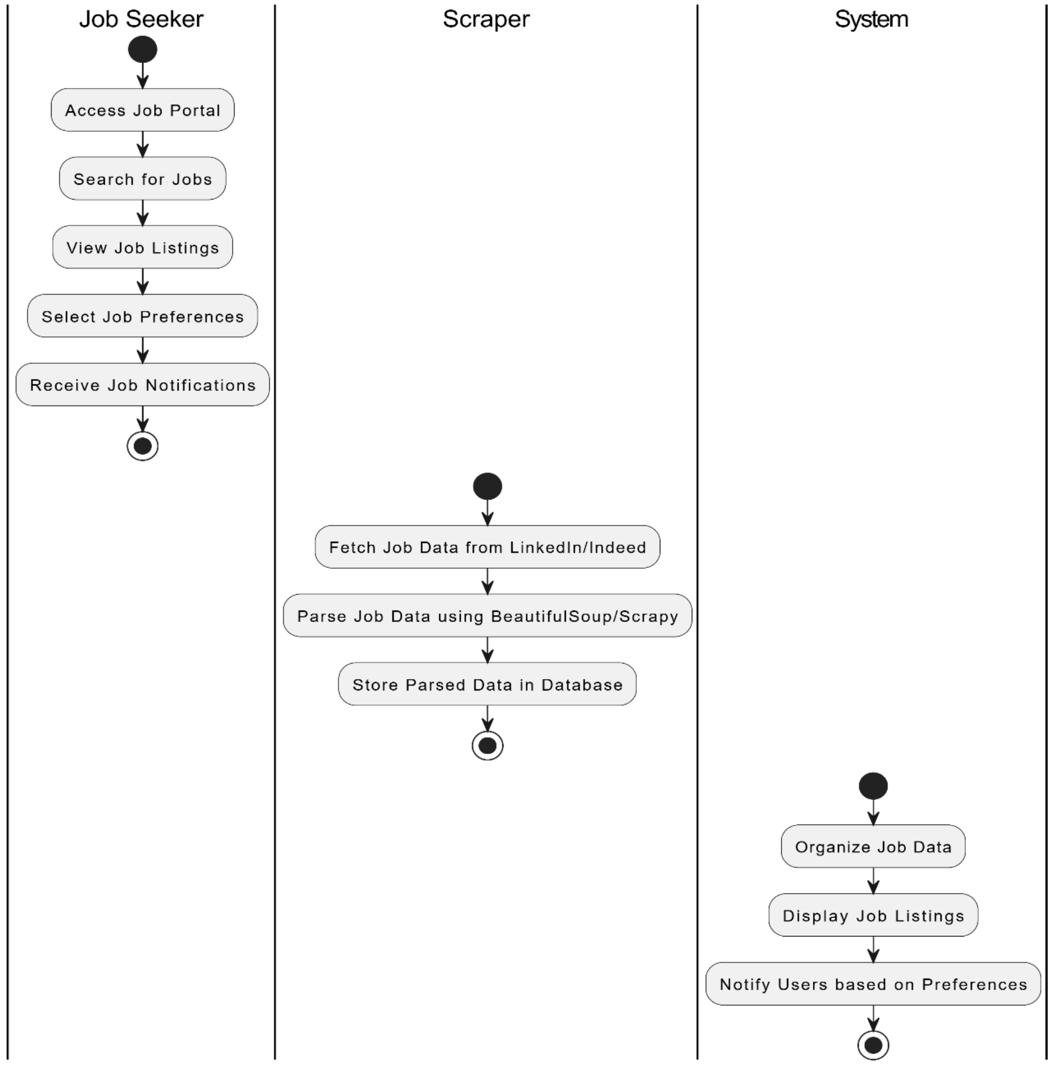


2.Use Case diagram: Use Case diagrams identify the functionality provided by the system (use cases), the users who interact with the system (actors), and the association between the users and the functionality

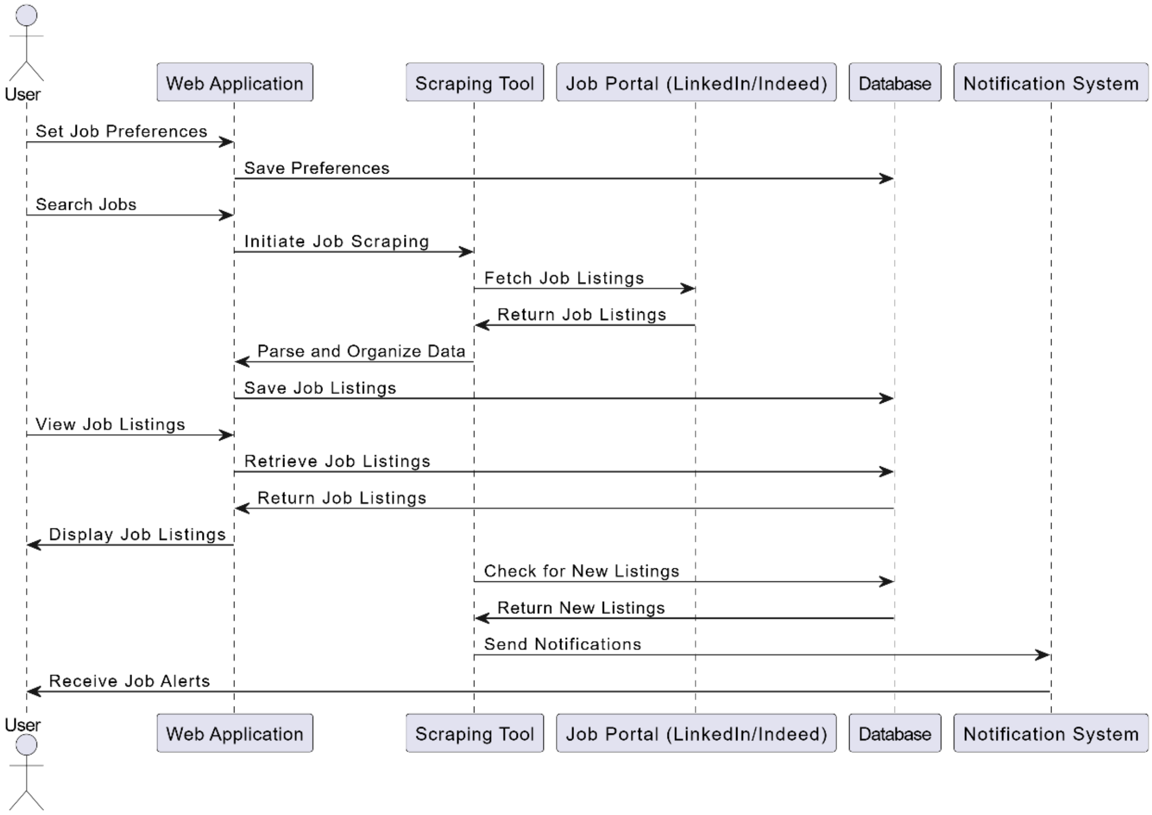


.

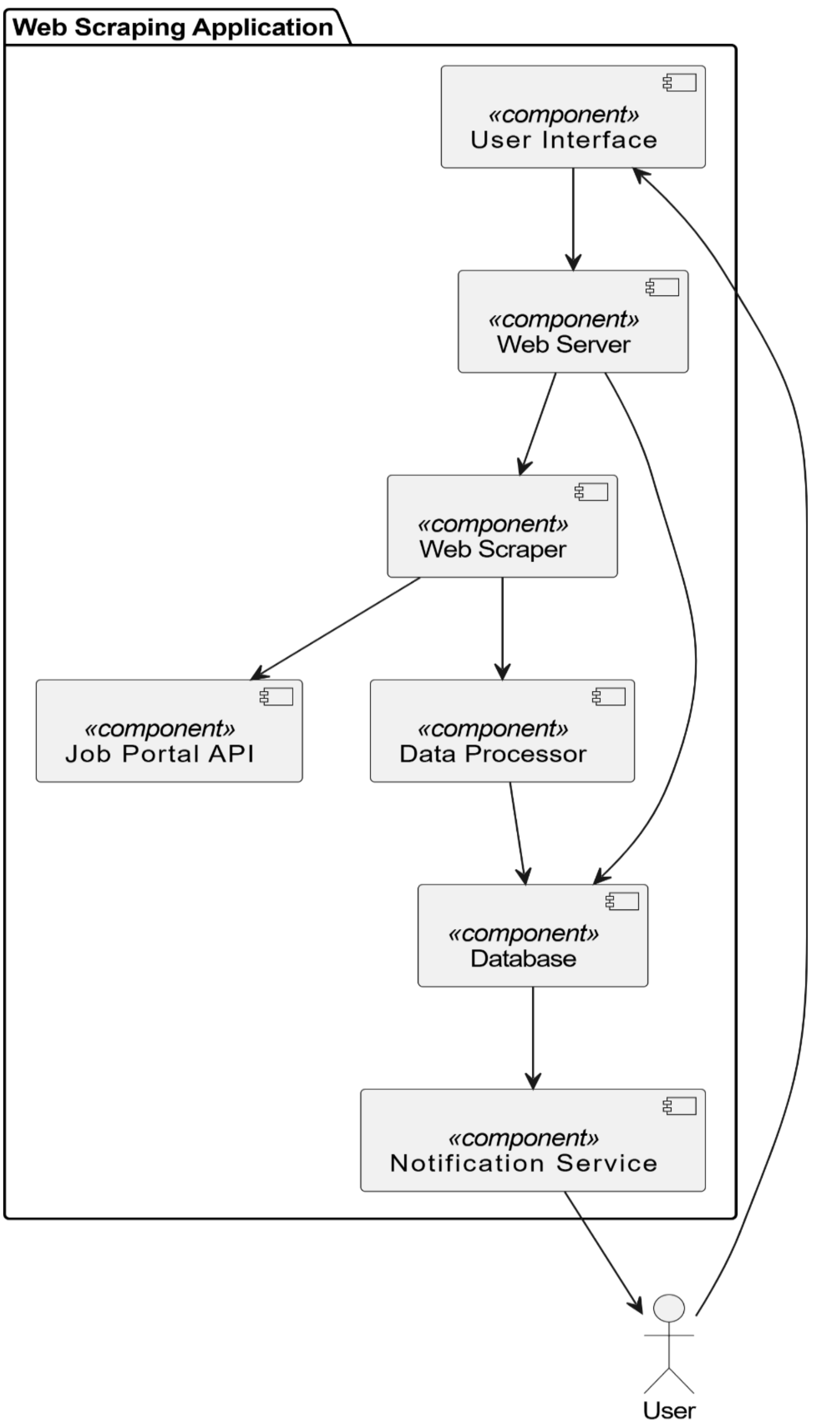
3.Activity Diagram: Activity diagrams are used to document workflows in a system, from the business level down to the operational level. The general purpose of Activity diagrams is to focus on flows driven by internal processing vs. external events.


Job Seeker
Scraper
System
Access Job Portal
Search for Jobs
View Job Listings
Select Job Preferences
Receive Job Notifications
Fetch Job Data from LinkedIn/Indeed
Parse Job Data using BeautifulSoup/Scrapy
Store Parsed Data in Database
Organize Job Data
Display Job Listings
Notify Users based on Preferences

4.Sequence Diagram: Sequence diagrams document the interactions between classes to achieve a result, such as a use case. The Sequence diagram lists objects horizontally, and time vertically, and models these messages over time. It showin the sequence of messages exchanged between them over time. Below is a basic description of how a sequence diagram could be structured for your job-scraping web application.



5.Component Diagram: A component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.



7.IMPLEMENTATION

7.1 Algorithm Used:

1. Initialize Preferences: Collect user preferences for job listings (e.g., job title, location).

2. Set URLs: Define target URLs for job portals like LinkedIn and Indeed based on user preferences.

3. Fetch HTML Content: Use Python libraries (BeautifulSoup, Scrapy) to send HTTP requests and fetch HTML content from the URLs.

4. Parse HTML: Use BeautifulSoup to parse the HTML content and identify relevant elements (e.g., job titles, descriptions, locations).

5. Extract Data: Extract job details (title, description, location, company, application deadline) from parsed HTML.

6. Store Data: Save the extracted job data into a structured format, such as a database or CSV file.

7. Filter Results: Filter job listings based on user preferences (if additional filtering is needed).

8. Notify Users: Use PHP to send notifications to users about new job listings via email or push notifications.

9. Update Data: Schedule regular scraping intervals to update the job listings and notify users of new opportunities.

10. Error Handling: Implement error handling for failed requests or parsing issues, and log errors for debugging.

7.2 Sample Code:

1 PHP Script for User Login:

```
<!doctype html>

<html lang="en">

<?php

include '../constants/settings.php';

include 'constants/check-login.php';

if ($user_online == "true") {

if ($myrole == "employer") {

}else{

header("location:../");

}

}else{

header("location:../");

}

?>

if($_SERVER["REQUEST_METHOD"]== "POST" &&
isset($_SESSION['user_id']) && $_SESSION['role'] == 'employer')
```

```
{ $job_title = $_POST['job_title'];

 $job_description = $_POST['job_description']; $company_name =
$_POST['company_name'];

 $job_location = $_POST['job_location'];

 $salary = $_POST['salary'];

 $sql = "INSERT INTO jobs (job_title, job_description, company_name,
job_location, salary) VALUES ('$job_title', '$job_description',
'$company_name', '$job_location', '$salary')";

if (mysqli_query($conn, $sql)) { echo "Job posted successfully."; }

 else{

echo"Error:".$sql."
" . mysqli_error($conn); }
```

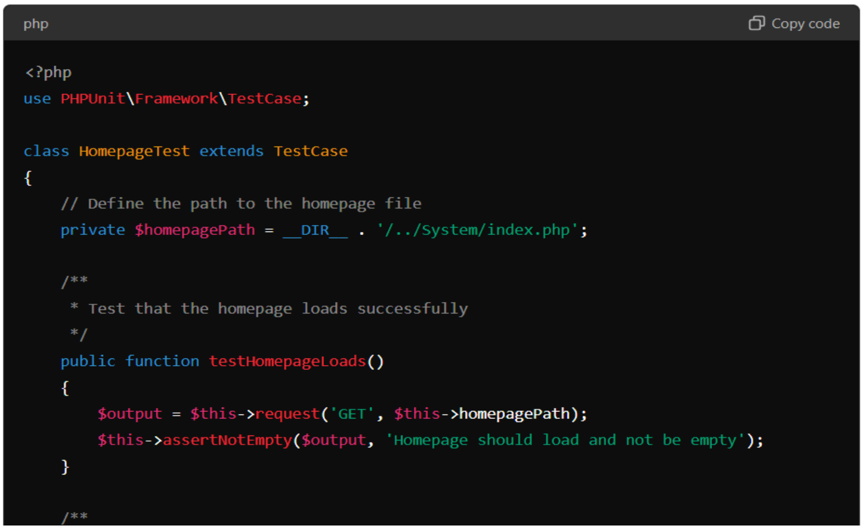
8.TESTING

Testing a web scraping and interface project involves several key mechanisms to ensure functionality, performance, and security. Here's a comprehensive overview of the testing mechanisms suitable for an online job portal project that uses Python for web scraping and PHP for the interface:

1. Unit Testing

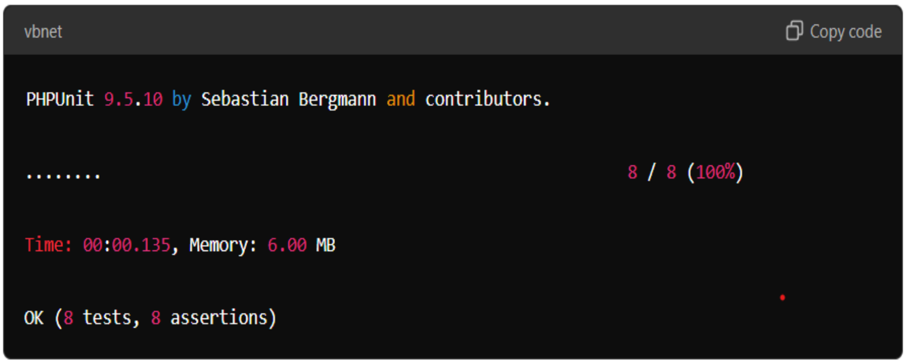
Python Web Scraping Script:

- **Purpose:** Verify the functionality of each function or component in the scraping script.
- **Tools:** `unittest` or `pytest` in Python.
- **Source code:**



```php
<?php
use PHPUnit\Framework\TestCase;

class HomepageTest extends TestCase
{
    // Define the path to the homepage file
    private $homepagePath = __DIR__ . '/../System/index.php';

    /**
     * Test that the homepage loads successfully
     */
    public function testHomepageLoads()
    {
        $output = $this->request('GET', $this->homepagePath);
        $this->assertNotEmpty($output, 'Homepage should load and not be empty');
    }

    /**
```

Output:

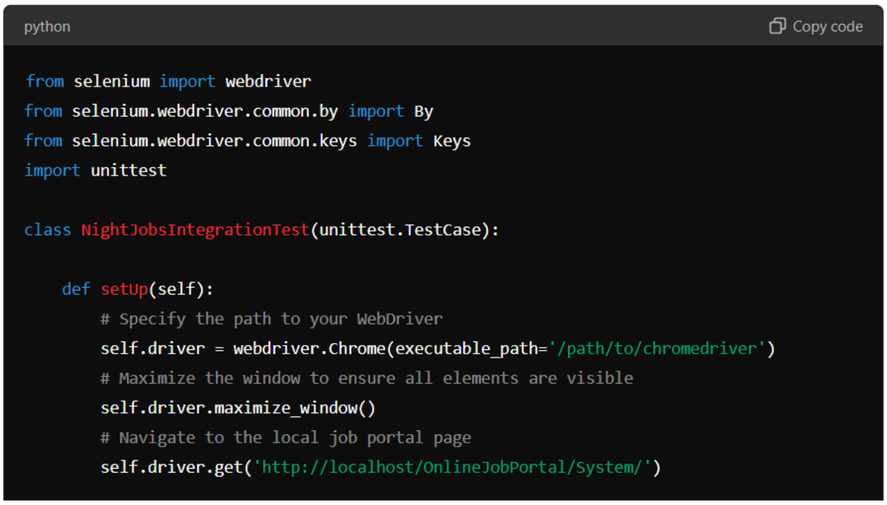


```
PHPUnit 9.5.10 by Sebastian Bergmann and contributors.

........                                                    8 / 8 (100%)

Time: 00:00.135, Memory: 6.00 MB

OK (8 tests, 8 assertions)
```

2. Integration Testing

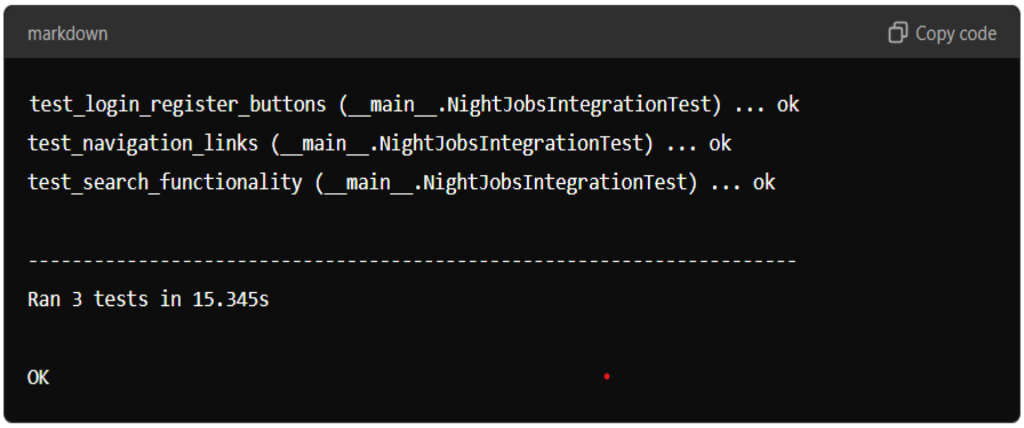
Integration of Python and PHP Components:

- **Purpose:** Ensure that the Python script and the PHP interface work together seamlessly.
- **Source Code**



```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
import unittest

class NightJobsIntegrationTest(unittest.TestCase):

    def setUp(self):
        # Specify the path to your WebDriver
        self.driver = webdriver.Chrome(executable_path='/path/to/chromedriver')
        # Maximize the window to ensure all elements are visible
        self.driver.maximize_window()
        # Navigate to the local job portal page
        self.driver.get('http://localhost/OnlineJobPortal/System/')
```

Output:

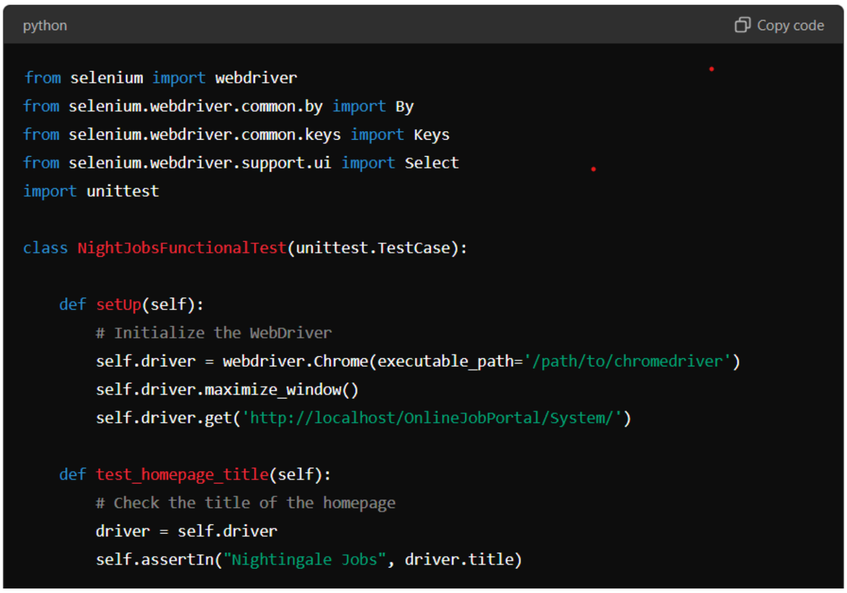


```
test_login_register_buttons (__main__.NightJobsIntegrationTest) ... ok
test_navigation_links (__main__.NightJobsIntegrationTest) ... ok
test_search_functionality (__main__.NightJobsIntegrationTest) ... ok

----------------------------------------------------------------------
Ran 3 tests in 15.345s

OK
```

3. Functional Testing

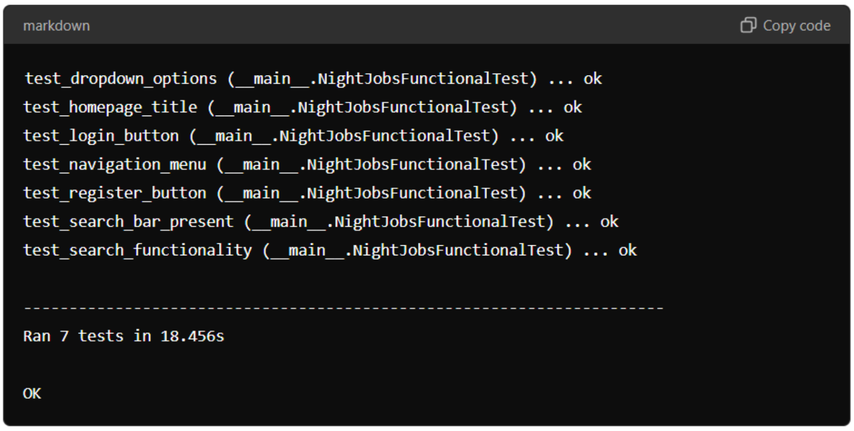
End-to-End Testing of the Job Portal:

- **Purpose:** Ensure the system behaves as expected from the user's perspective.
- **Tools:** Selenium (for automating browser interactions).
- **Source code:**



```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.support.ui import Select
import unittest

class NightJobsFunctionalTest(unittest.TestCase):

    def setUp(self):
        # Initialize the WebDriver
        self.driver = webdriver.Chrome(executable_path='/path/to/chromedriver')
        self.driver.maximize_window()
        self.driver.get('http://localhost/OnlineJobPortal/System/')

    def test_homepage_title(self):
        # Check the title of the homepage
        driver = self.driver
        self.assertIn("Nightingale Jobs", driver.title)
```

- **:Output:**

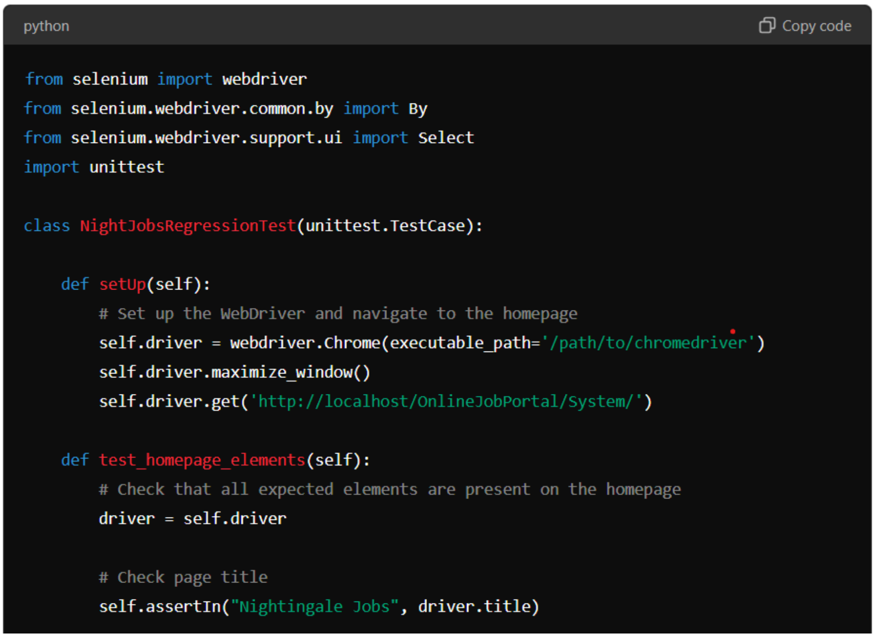


```
test_dropdown_options (__main__.NightJobsFunctionalTest) ... ok
test_homepage_title (__main__.NightJobsFunctionalTest) ... ok
test_login_button (__main__.NightJobsFunctionalTest) ... ok
test_navigation_menu (__main__.NightJobsFunctionalTest) ... ok
test_register_button (__main__.NightJobsFunctionalTest) ... ok
test_search_bar_present (__main__.NightJobsFunctionalTest) ... ok
test_search_functionality (__main__.NightJobsFunctionalTest) ... ok

----------------------------------------------------------------------
Ran 7 tests in 18.456s

OK
```

4. Regression Testing

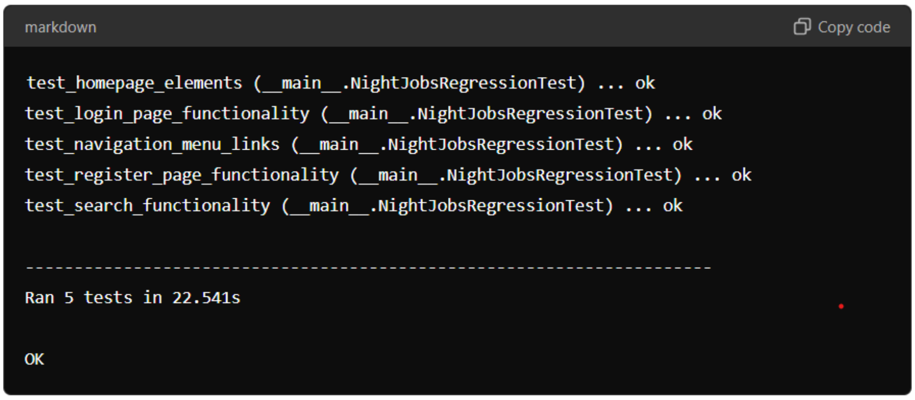
Ensuring New Updates Do Not Break Existing Features:

Source Code:



```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import Select
import unittest

class NightJobsRegressionTest(unittest.TestCase):

    def setUp(self):
        # Set up the WebDriver and navigate to the homepage
        self.driver = webdriver.Chrome(executable_path='/path/to/chromedriver')
        self.driver.maximize_window()
        self.driver.get('http://localhost/OnlineJobPortal/System/')

    def test_homepage_elements(self):
        # Check that all expected elements are present on the homepage
        driver = self.driver

        # Check page title
        self.assertIn("Nightingale Jobs", driver.title)
```

Output:



```
test_homepage_elements (__main__.NightJobsRegressionTest) ... ok
test_login_page_functionality (__main__.NightJobsRegressionTest) ... ok
test_navigation_menu_links (__main__.NightJobsRegressionTest) ... ok
test_register_page_functionality (__main__.NightJobsRegressionTest) ... ok
test_search_functionality (__main__.NightJobsRegressionTest) ... ok

----------------------------------------------------------------------
Ran 5 tests in 22.541s

OK
```

9.RESULTS

The Careerscope Connect system delivers various outputs tailored to different user roles, enhancing their experience and functionality within the platform:

Admin Features:

- Job Application Management: Admins can accept or reject job applications and delete job postings as needed.

- Comprehensive Dashboard: Provides a centralized view for managing user activities and system configurations.

- Data Management: Enables control over user roles, permissions, and data integrity. Employer Features:

- Job Posting: Employers can post job listings with detailed descriptions and specifications.

- Applicant Screening: Tools to review and manage job applicants efficiently.

- Dashboard Customization: Tailored navigation bars for streamlined access to employer-specific features.

Jobseeker Features:

- Job Search: Users can browse available job listings based on various criteria.

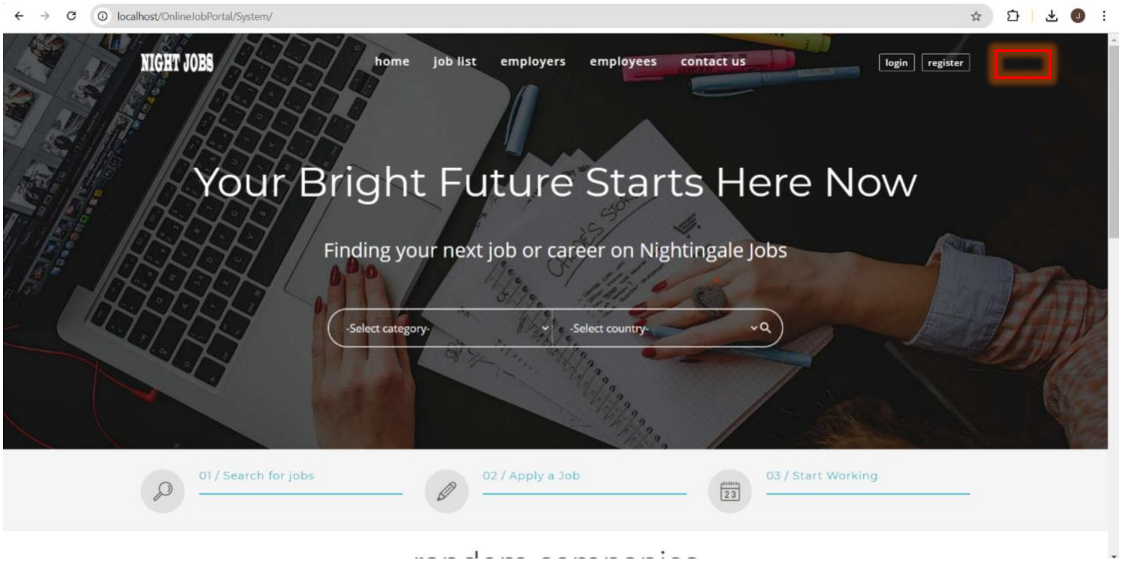
- Application Management: Facilitates applying for jobs directly through the platform.

- Profile Management: Allows jobseekers to maintain and update their profiles easily.

Interface Screenshot

Admin

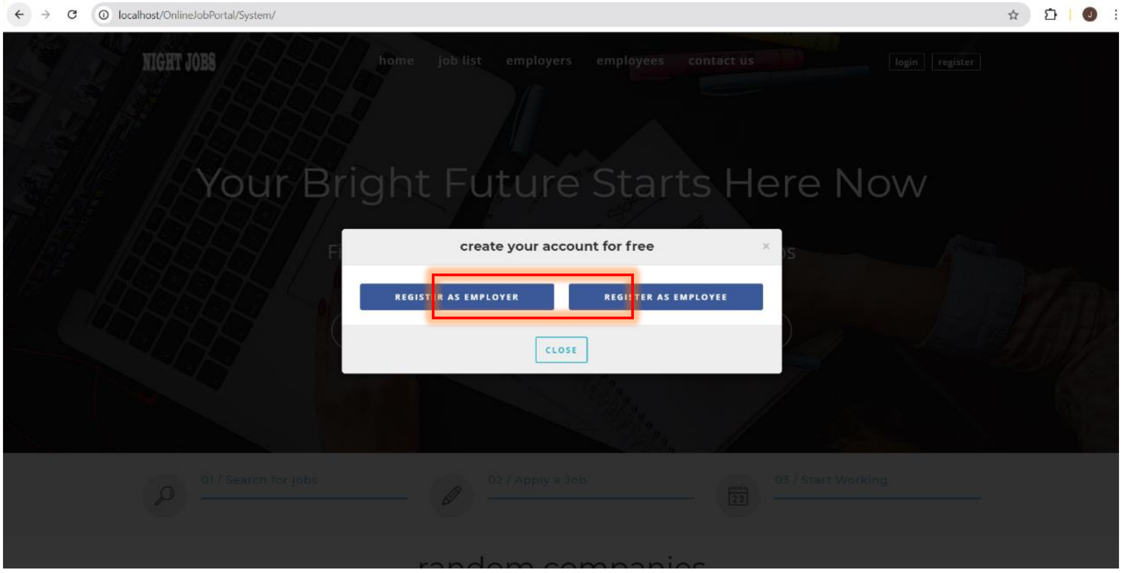
Home Page



Your Bright Future Starts Here Now

Finding your next job or career on Nightingale Jobs

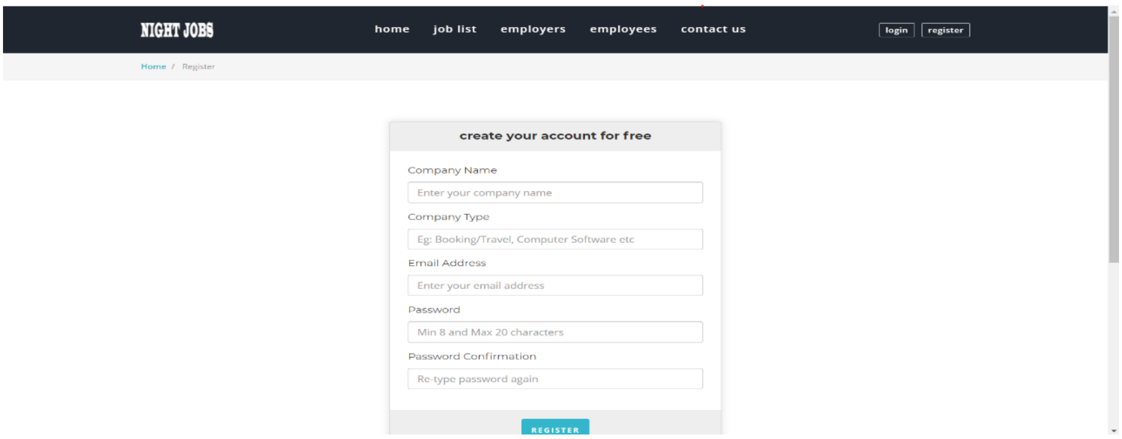
Register as Employer



Your Bright Future Starts Here Now

create your account for free

Registration_Page



create your account for free

Company Name: Enter your company name

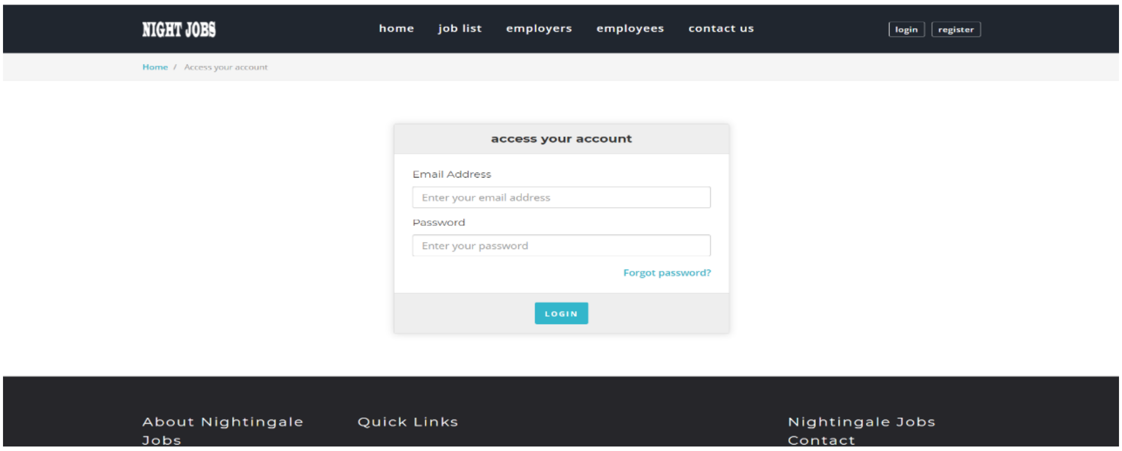
Company Type: Eg: Booking/Travel, Computer Software etc

Email Address: Enter your email address

Password: Min 8 and Max 20 characters

Password Confirmation: Re-type password again

Login Page

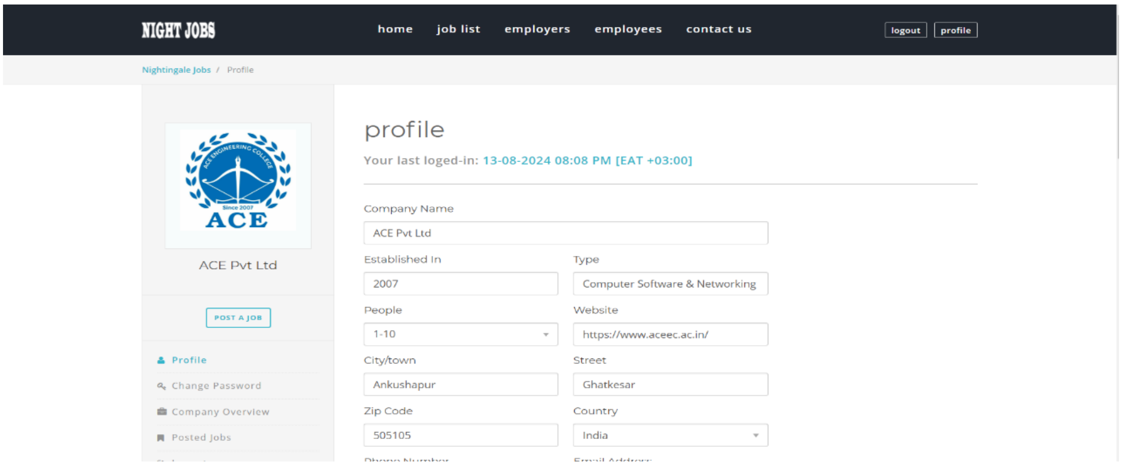


access your account

Email Address: Enter your email address

Password: Enter your password

Recruiter_Profile



profile

Your last loged-in: 13-08-2024 08:08 PM [EAT +03:00]

Company Name: ACE Pvt Ltd

Established In: 2007

Type: Computer Software & Networking

People: 1-10

Website: https://www.aceec.ac.in/

City/town: Ankushapur

Street: Ghatkesar

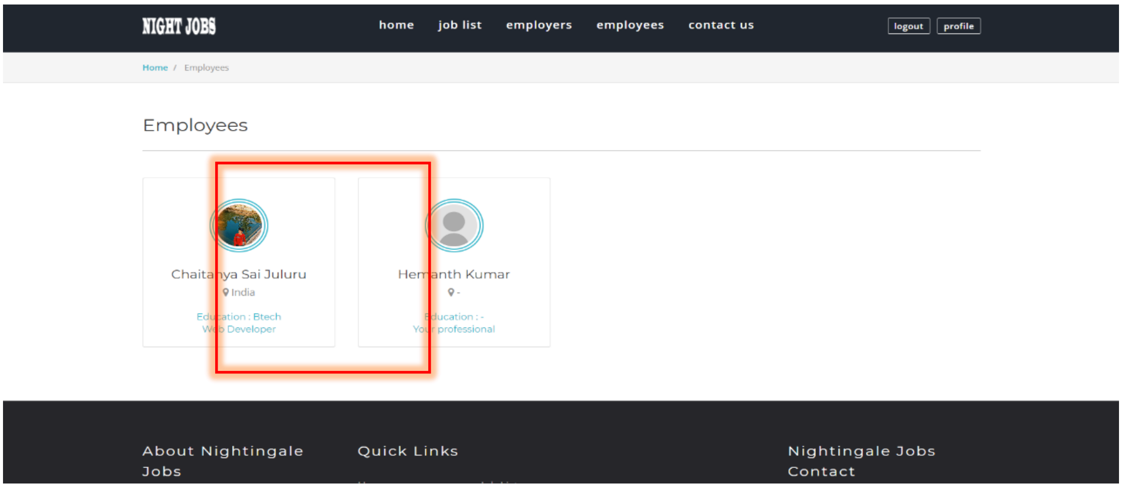
Zip Code: 505105

Country: India

Phone Number

Email Address

Job Seekers List

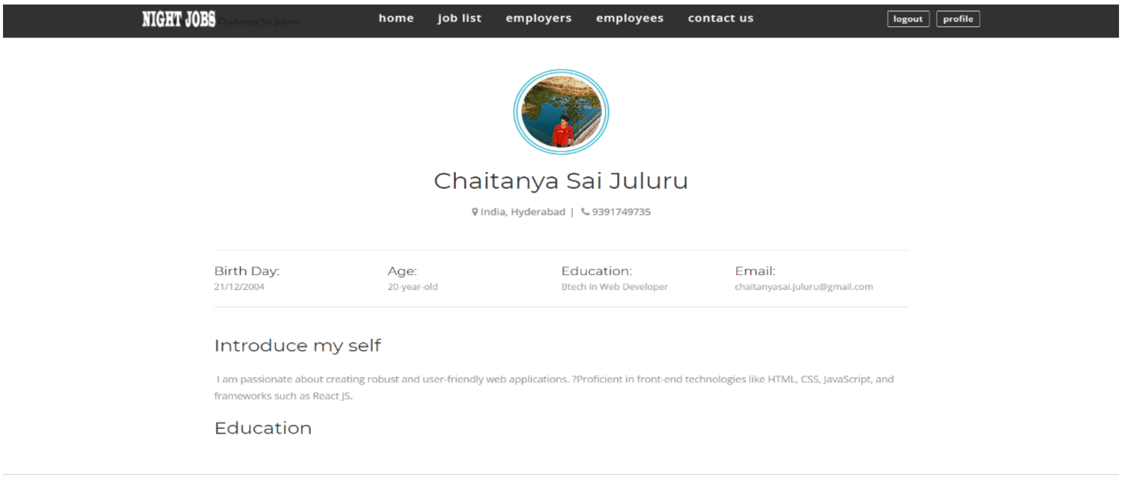


Employees

Chaitanya Sai Juluru
India
Education : Btech
Web Developer

Hemanth Kumar
-
Education : -
Your professional

Job Seeker Profile



Chaitanya Sai Juluru

India, Hyderabad | 9391749735

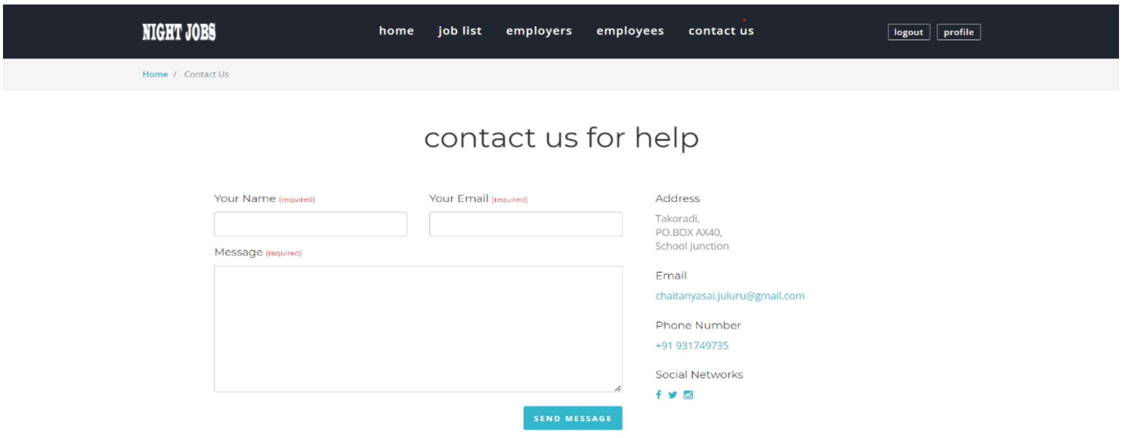
Birth Day:	Age:	Education:	Email:
21/12/2004	20-year-old	Btech in Web Developer	chaitanyasai.juluru@gmail.com

Introduce my self

I am passionate about creating robust and user-friendly web applications. ?Proficient in front-end technologies like HTML, CSS, JavaScript, and frameworks such as React JS.

Education

Contact_Us_Page



contact us for help

Your Name (required)

Your Email (required)

Message (required)

Address

Takoradi,
PO.BOX AX40,
School Junction

Email

chaitanyasai.juluru@gmail.com

Phone Number

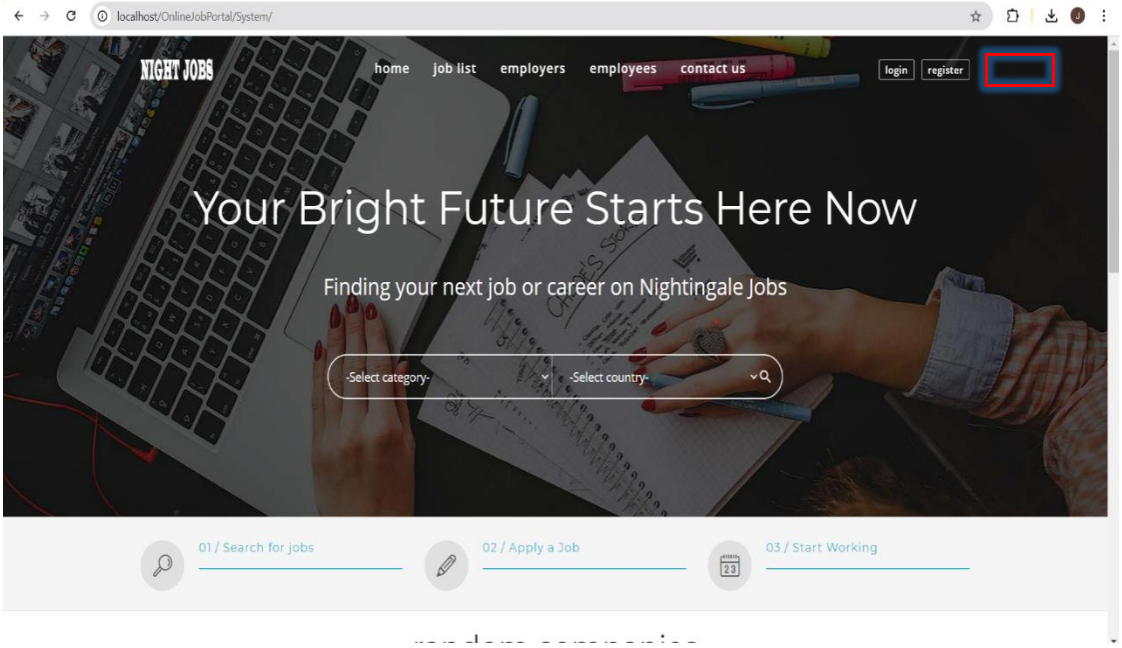
+91 931749735

Social Networks

Interface Screenshots

Job seekers

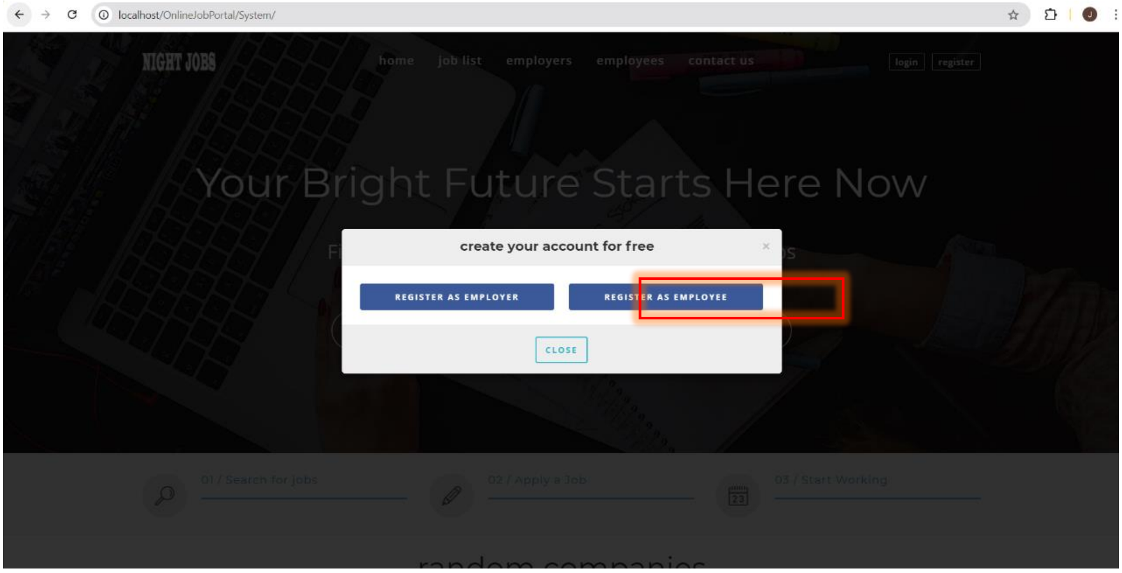
Home Page



Your Bright Future Starts Here Now

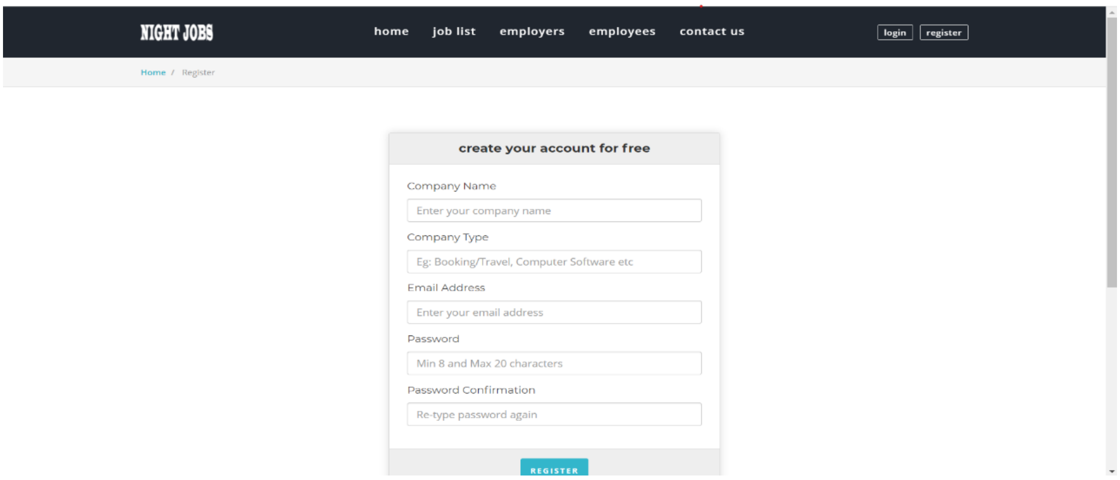
Finding your next job or career on Nightingale Jobs

Register as Employee



create your account for free

Registration Page



create your account for free

Company Name

Enter your company name

Company Type

Eg: Booking/Travel, Computer Software etc

Email Address

Enter your email address

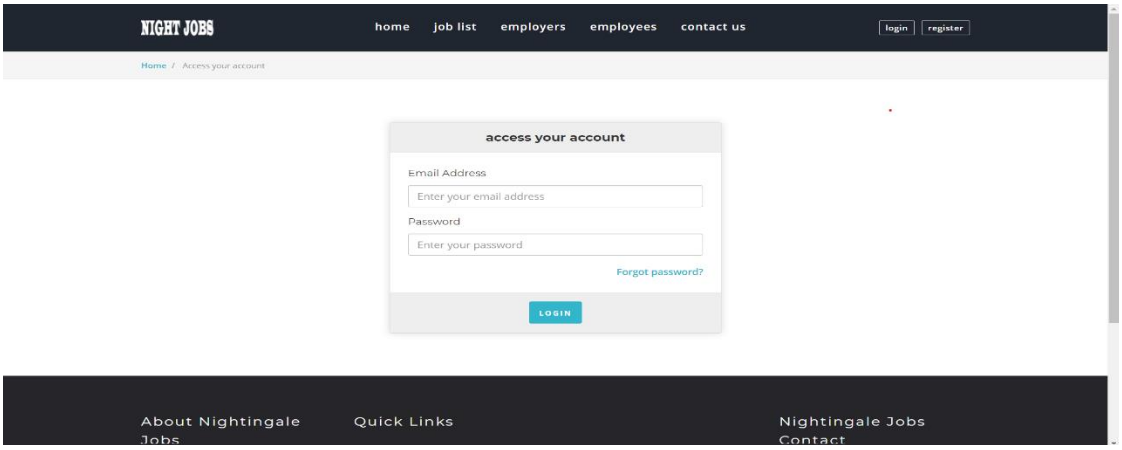
Password

Min 8 and Max 20 characters

Password Confirmation

Re-type password again

Login Page



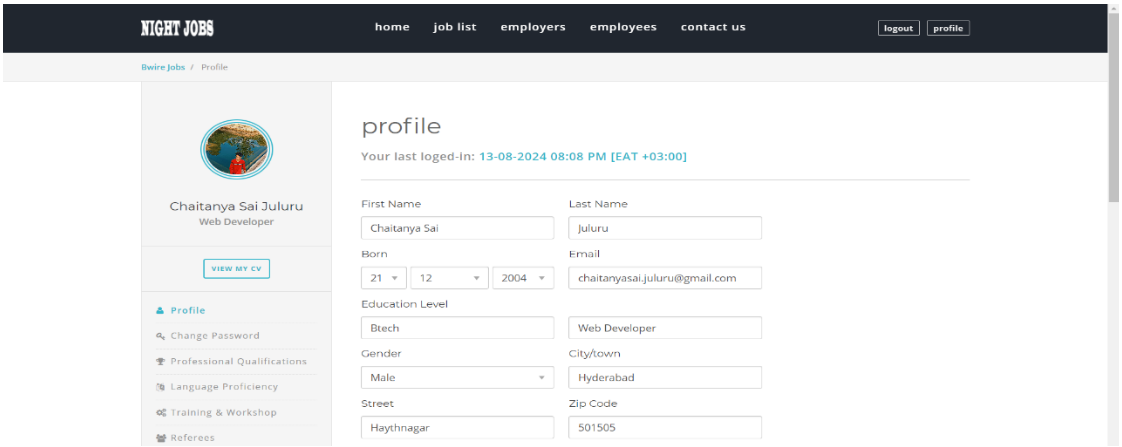
Email Address

Enter your email address

Password

Enter your password

UserProfile



Chaitanya Sai Juluru

Web Developer

profile

Your last loged-in: 13-08-2024 08:08 PM [EAT +03:00]

First Name: Chaitanya Sai

Last Name: Juluru

Born: 21 12 2004

Email: chaitanyasai.juluru@gmail.com

Education Level: Btech

Web Developer

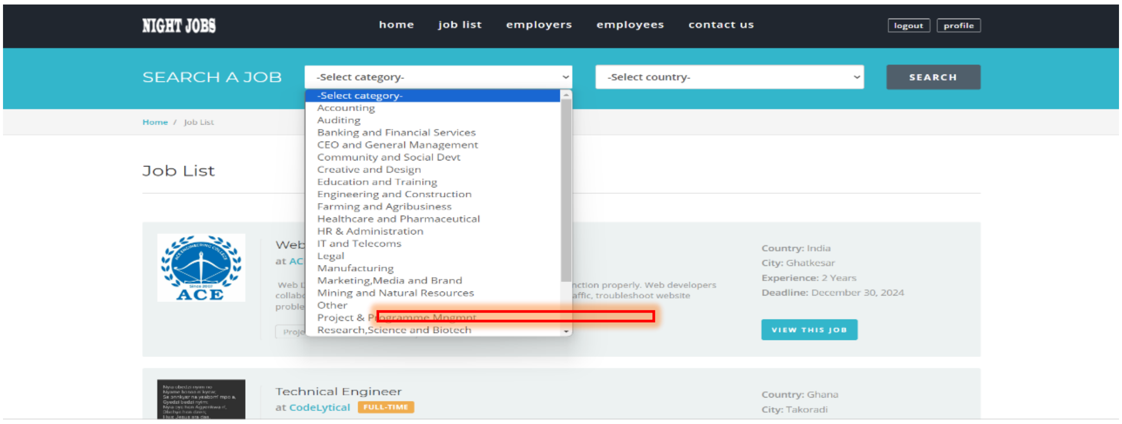
Gender: Male

City/town: Hyderabad

Street: Haythnagar

Zip Code: 501505

Job seeker can Search for category



-Select category-
Accounting
Auditing
Banking and Financial Services
CEO and General Management
Community and Social Devt
Creative and Design
Education and Training
Engineering and Construction
Farming and Agribusiness
Healthcare and Pharmaceutical
HR & Administration
IT and Telecoms
Legal
Manufacturing
Marketing,Media and Brand
Mining and Natural Resources
Other
Project & Programme Mngmnt
Research,Science and Biotech

Job List

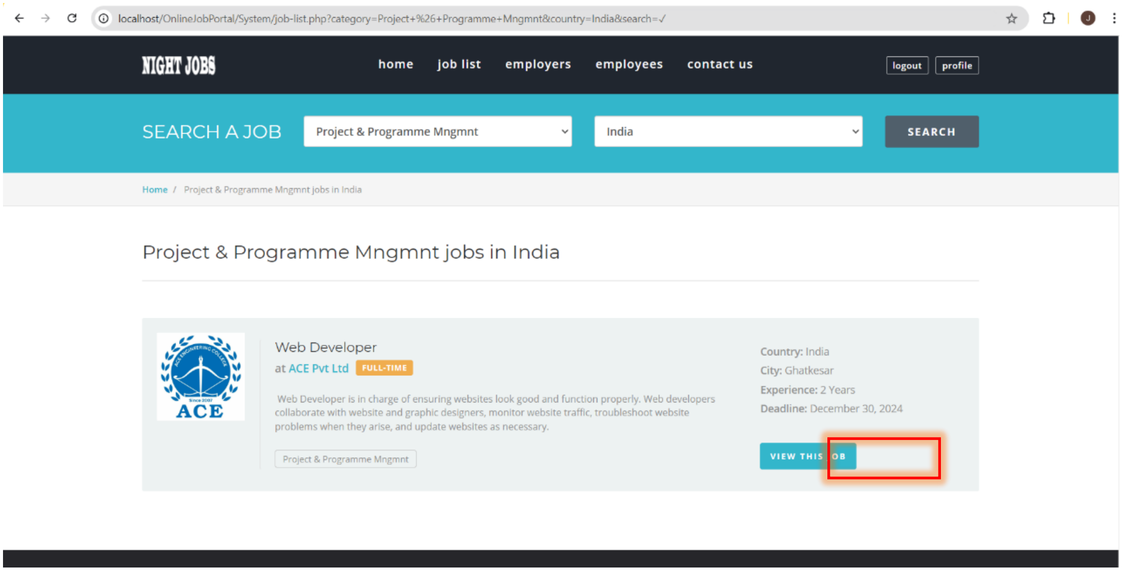
Web
at AC

Web D... collab... proble... ...nction properly. Web developers ...affic, troubleshoot website

Country: India
City: Ghatkesar
Experience: 2 Years
Deadline: December 30, 2024

Technical Engineer
at CodeLytical FULL-TIME

Country: Ghana
City: Takoradi



localhost/OnlineJobPortal/System/job-list.php?category=Project+%26+Programme+Mngmnt&country=India&search=✓

Project & Programme Mngmnt

India

Home / Project & Programme Mngmnt jobs in India

Project & Programme Mngmnt jobs in India

Web Developer

at ACE Pvt Ltd FULL-TIME

Web Developer is in charge of ensuring websites look good and function properly. Web developers collaborate with website and graphic designers, monitor website traffic, troubleshoot website problems when they arise, and update websites as necessary.

Project & Programme Mngmnt

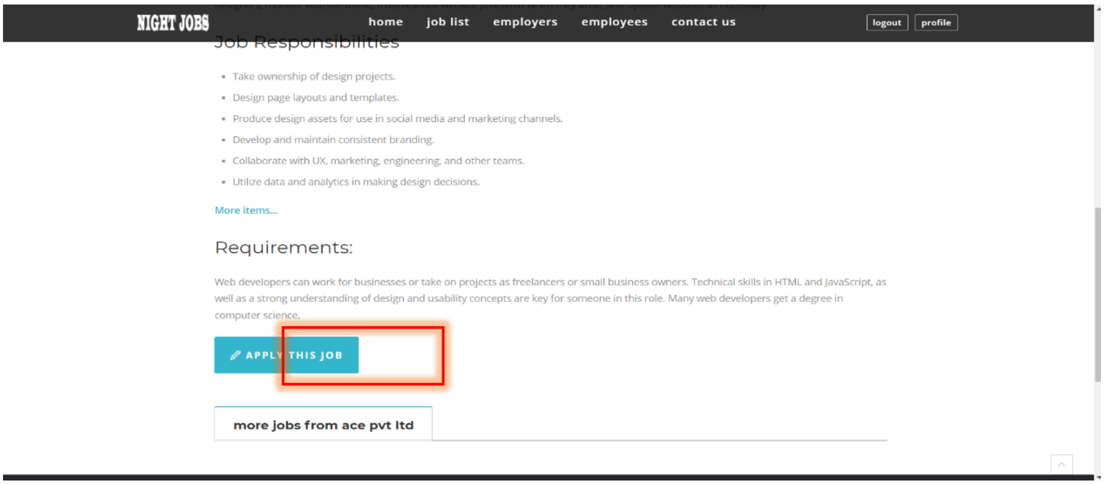
Country: India

City: Ghatkesar

Experience: 2 Years

Deadline: December 30, 2024

User_can_apply_the_job



Job Responsibilities

- Take ownership of design projects.
- Design page layouts and templates.
- Produce design assets for use in social media and marketing channels.
- Develop and maintain consistent branding.
- Collaborate with UX, marketing, engineering, and other teams.
- Utilize data and analytics in making design decisions.

More items...

Requirements:

Web developers can work for businesses or take on projects as freelancers or small business owners. Technical skills in HTML and JavaScript, as well as a strong understanding of design and usability concepts are key for someone in this role. Many web developers get a degree in computer science.

10.CONCLUSION

The increasing demand for job opportunities necessitates efficient data retrieval mechanisms from online job portals such as LinkedIn and Indeed. This project successfully demonstrates the design and development of a web application that systematically retrieves and organizes job postings from various web portals, providing users with relevant job opportunities based on their preferences.

The web scraping application, utilizing Python libraries such as BeautifulSoup, Scrapy, and PHP, effectively serves as the backend for data extraction. HTML and CSS are employed to structure and present the extracted data, including essential details such as job titles, descriptions, locations, and application deadlines. The automated process of gathering job listings not only streamlines the job search process for users but also aids recruiters in making data-driven decisions.

By automating the retrieval and organization of job listings, this project enhances the efficiency and effectiveness of the job market. The significant role of web scraping in accessing valuable information from online sources is underscored, paving the way for improved user experiences and informed decision-making. This application stands as a testament to the power of modern web scraping technologies in meeting the dynamic demands of today's job market, ultimately benefiting job seekers and recruiters alike.

FUTURE WORK

1.Enhanced Data Filtering and Personalization: Future developments could focus on incorporating advanced data filtering techniques and machine learning algorithms to provide more personalized job recommendations. By analyzing users' past interactions, preferences, and profile details, the application could offer highly tailored job opportunities that align with individual career goals. This enhancement would significantly improve user experience by delivering more relevant and targeted job listings.

2.Real-time Notifications and Alerts: Expanding the application's capabilities to include real-time notifications and alerts would further enhance its utility. Users could receive immediate updates on new job postings that match their criteria, ensuring they are among the first to apply. This could be achieved by integrating APIs that monitor job portals continuously and trigger notifications through email, SMS, or in-app alerts, thus making the job search process more proactive and time-sensitive.

3.Cross-Platform Compatibility and Integration: To broaden the application's accessibility, future work could involve developing a mobile-friendly version or a dedicated mobile app, allowing users to search for jobs and receive notifications on the go. Additionally, integrating the application with professional networking sites like LinkedIn or job search engines such as Indeed could enhance its functionality by enabling users to apply directly from the app and sync their profiles across platforms. This cross-platform approach would make the job search process more seamless and efficient for users

REFERENCES.

[1] Literature review on online job portal

https://ijarcce.com/upload/2017/may-17/IJARCCE%2019.pdf

[2] Pinjari, M., De, N., Kokne, R., Siddiqui, A., & Chitre, D. (2019). Online Job Portal. International Research Journal of Engineering and Technology.

[3] Mithun, G. (2020). A Project Report On Job Portal (Doctoral dissertation, CMR Institute of Technology.Bangalore).

[4] Khan, M. S., & Khan, M. S. (2015). Online job portal (Doctoral dissertation, University of Management and Technology Lahore).

[5] Chowdhury, A. R., Areias, A. C., Imaizumi, S., Nomura, S., & Yamauchi, F. (2018). Reflections of employers' gender preferences in job ads in India: an analysis of online job portal data. World Bank Policy Research Working Paper, (8379).