St Joseph’s College of
Engineering,Chennai-600119

AI RESUME ANALYZER

A tool for resume analysis, prediction and recommendations

A MINI PROJECT

BACHELOR OF TECHNOLOGY-IT

BY

ANGELA SHYLYN P

Reg no: 312323205019

ABSTRACT

The aim of this project is to design and develop a tool that results into an easy and helpful solution for applicants as well as recruiters

“AI RESUME ANALYZER” which parses information from a resume using natural language processing, finds the keywords, cluster them onto sectors based on their keywords

And lastly show the recommendation, prediction, analytics to the applicant / recruiter based on keyword matching.

TABLE OF CONTENTS

INTRODUCTION

1.1. Background

Corporate companies and recruitment agencies process numerous resumes daily. This is no task for humans. An automated intelligent system is required which can take out all the vital information from the unstructured resumes and transform all of them to a common structured format which can then be ranked for a specific job position.

Parsed information includes (name, email address, phone number, work experiences, education, hobbies, interests, achievements, certifications, projects) keywords and finally the cluster of the resume (ex: Web Development, Data Science etc.). The parsed information is then stored in a database (MySQL in this case) for later use.

Unlike other unstructured data (ex: email body, web page contents, etc.), resumes are a bit structured. Information is stored in discrete sets. Each set contains data about the person's contact, work experience or education details. In spite of this, resumes are difficult to parse. This is because they vary in types of information, their order, writing style, etc. To parse the data from different kinds of resumes effectively and efficiently, the model must not rely on the order or type of data.

To solve this tedious process our tool comes into action which makes the process fast, easy and reliable. Using NLP Techniques, it extracts keywords from the resume and use it for predictions, recommendation and analytical representation.

1.2. Objectives

The aim is to design and develop a model that can parse information from unstructured (pdf) data, and transform it to JSON for furthermore processing

A tool that analyses applicants resume and transform it to a structured JSON format, using parsing techniques and some programming fundamentals

Which can be used by any organization (company/college/individual user) that handles resume screening process

To keep a track of all records into database for further admin side analytics. And also, to provides tips and recommendations based on applicants resume

1.3. Purpose and Scope

1.3.1. Purpose

i. Research shows that 90% of all CVs/ Resumes are checked for much less than 2 minutes via the employers.

ii. This implies that in a maximum of the instances recruiters simply study the bits of critical components or the points of interest within the Resumes and ignores the rest.

iii. Therefore, the first goal was to make a tool that covers all the parts in a resume and keeps a track of all records within minimal time span

iv. To make it applicant's friendly and supportive by providing them predictions and recommendations

v. To make it recruiter friendly by providing them user data and export it to csv, and also to provide insights and analytics.

1.3.2. Scope

i. It can be used for getting all the resume data into a structured tabular format and csv as well, so that the organization can use those data for analytics purposes

ii. By providing recommendations, predictions and overall score user can improve their resume and can keep on testing it on our tool

iii. And it can increase more traffic to our tool because of user section

iv. It can be used by colleges to get insight of students and their resume before placements

v. Also, to get analytics for roles which users are mostly looking for

vi. To improve this tool by getting feedbacks

SYSTEM ANALYSIS

2.1. Present System

The process of hiring has evolved over the period of time. In the first-generation hiring model, the companies would advertise their vacancies on newspapers and television. The applicants would send in their resumes via post and their resumes would be sorted manually. Once shortlisted, the hiring team would call the applicants for further rounds of interview.

Needless to say, this was a time-consuming procedure. But the industries started growing and so did the hiring needs. Hence the companies started outsourcing their hiring process. Hiring consultancies came into existence. These agencies required the applicants to upload their resumes on their websites in particular formats. The agencies would then go through the structured data and shortlist candidates for the company.

This process had a major drawback. There were numerous agencies and each had their own unique format. To overcome all the above problems an intelligent algorithm was required which could parse information from any unstructured resumes, sort it based on the clusters and rank it finally.

2.1.1. Limitations of present system

This is no task for humans and time consuming.

It is challenging task to handle resume manually.

Clashes due to their own unique format.

Requires individual review of each resume from hiring managers.

The same amount of time and effort is often expelled for candidates who are qualified as the ones who are.

2.2. Proposed System

The proposed system “AI RESUME ANALYZER” is an applicant cum recruiter-based solution which can be widely used by any organization to analyze and get insights of a resume

The model uses natural language processing to understand the resume and then parse the information from it.

Once information is parsed it is stored in the database.

Quite productive for applicants because it gives predictions, tips and recommendations based on their resume information

System works proper when the uploaded resume is in traditional chronological format

Insightful for admin/recruiter due to its powerful analytics and informative data which is fetched from user/applicants resume

2.2.1. Advantages of proposed system

Tracks and Analyze Resume Based on Job Roles.

Fast, Safe, Real-time Predictions.

Does the task within less timespan

Provides more Efficient Review Overall.

Helpful and accurate for applicants as well as recruiters

2.3. Hardware Requirements

A Laptop / Desktop Connected with Internet

Minimum of 4GB ram or more

Single Network Connection

So that other devices within network can connect through network URL

2.4. Software Requirements

MySQL

Python

Browser (chrome recommended)

Text Editor (Visual Studio Code recommended)

2.5. Justification of selection technology

Frontend: -

- HTML5:-
  HTML is the standard markup language for Web pages. With HTML you can create your own website. CSS3: -
- CSS is the language we use to style an HTML document. CSS describes how HTML elements should be displayed. JavaScript: - JavaScript is the world's most popular programming language. JavaScript is the programming language of the Web. Streamlit: -
- Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science. In just a few minutes you can build and
- deploy powerful data apps.

Backend: -

- Python: -
  Python is a popular programming language. Python can be used on a server to create web applications.
- JSON: -
  JSON is a text format for storing and transporting data. JSON is "self-describing" and easy to understand

Database: -

- MySQL: -
  MySQL is free and open-source. And is a widely used relational database management system (RDBMS)

SYSTEM DESIGN

3.1. Module Division

This tool deals with three modules based on the working.

Client: -
- Fetching Location and Miscellaneous Data
- Using Parsing Techniques to fetch
  - Basic Info
  - Skills
  - Keywords
- Using logical programs, it will recommend
  - Skills that can be added
  - Predicted job role
  - Course and certificates
  - Resume tips and ideas
  - Overall Score
  - Interview & Resume tip videos

Admin: -
- Get all applicant's data into tabular format
- Download user's data into csv file
- View all saved uploaded pdf in Uploaded Resume folder
- Get user feedback and ratings
- Pie Charts for: -
  - Ratings
  - Predicted field / roles
  - Experience level
  - Resume score
  - User count
  - City
  - State
  - Country

Feedback: -
- Form filling
- Rating from 1 – 5
- Show overall ratings pie chart
- Past user comments history

3.2. Database Design

A pictorial representation of how the database and its table is

admin_cred

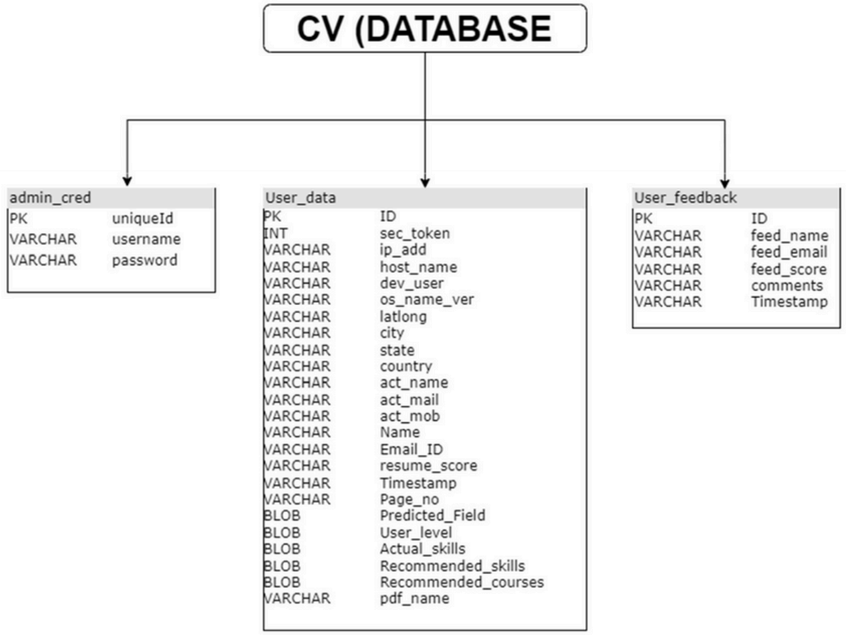
Stores data of user credentials

user_data

Stores data of parsed and fetched information from user and it's resume

user_feedback

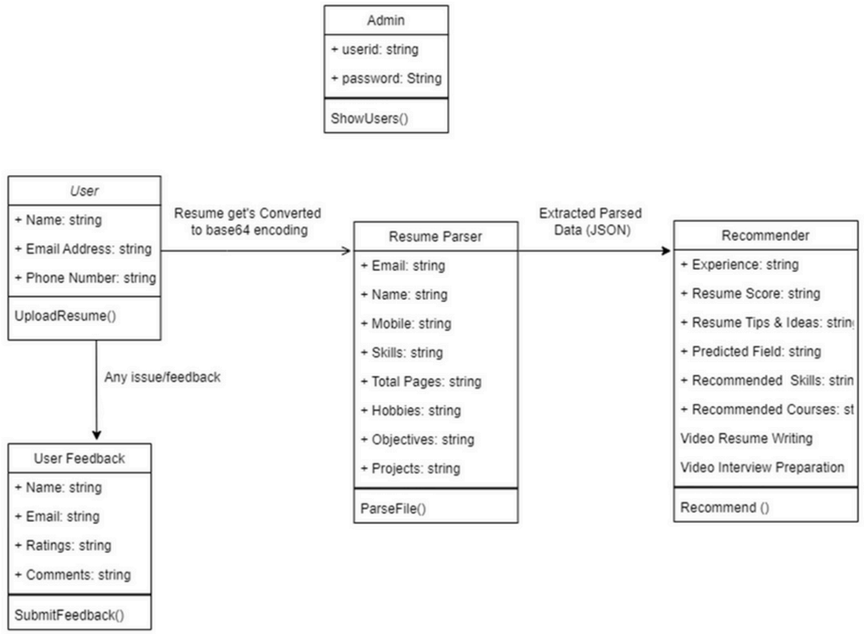
Stores feedback data provided by user's



Database Design

3.3. Class Diagram

A pictorial representation of how the process works using class diagram. The process starts by uploading resume from user. While uploading resume user has to provide their name, email-address, phone number A process will work behind the scene which will fetch Ip address and based on Ip address it will fetch user location details and some miscellaneous data also. After the resume is uploaded and saved to the root folder the parser will start parsing resume and convert those data into JSON format. The recommender will use those data for predicting experience, field of interest, resume overall score and also provides recommendations like skills, tips & ideas, courses, video (resume writing and interview preparation). After the process is done all the data gets stored into database



Class Diagram For Resume Analyzer

3.4. Event Table

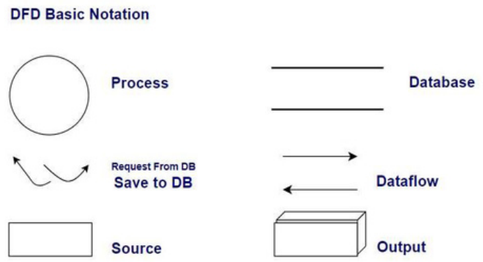
The below event table shows types of events that can be used with our tool

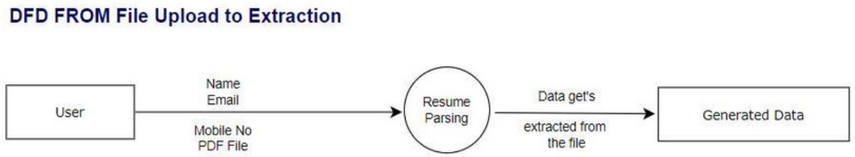
EVENT	TRIGGER	SOURCE	USE CASE	RESPONSE	DESTINATION
User Want's to analyze his resume	Upload Resume	User	Analyze Resume	Analyzed Precdicted Result	User
User Want's to give feedback about this tool	New Feedback	User	Make a review	Review Analyzed History	User
Admin wants to view user's data, customer feedback and analytics	Login as Admin	Admin	view available user's data, customer feedback and show analytics	Data Table's and pie charts	Admin

EVENT TABLE

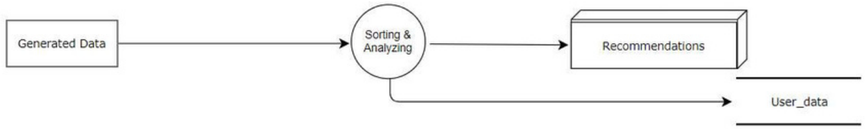
3.5. Data Flow Diagram

The below figures describe how the data flows through out the process

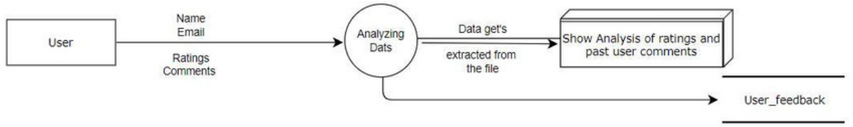




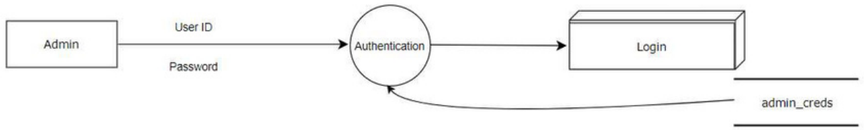
DFD FROM Extraction to Recommender



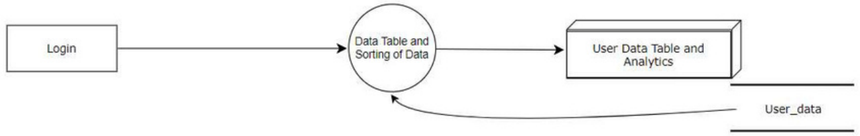
DFD FOR Feedback



DFD FOR Admin Login



DFD FOR Admin User Table and Data Visualizations



IMPLEMENTATION

Source Code : - https://github.com/deepakpadhi986/AI-Resume-Analyzer

Here are some of the sliced modules which were implemented to create this tool

To get downloadable link of user data into csv format

```
def get_table_download_link(df,filename,text):
    csv = df.to_csv(index=False)
    # bytes conversions
    b64 = base64.b64encode(csv.encode()).decode()
    href = f'<a href="data:file/csv;base64,{b64}" download="{filename}">{text}</a>'
    return href
```

Parser's code is not been added in this document because it is too long instead of that we added pdf reader

```
# Reads Pdf file and check_extractable
def pdf_reader(file):
    resource_manager = PDFResourceManager()
    fake_file_handle = io.StringIO()
    converter = TextConverter(resource_manager, fake_file_handle, laparams=LAParams())
    page_interpreter = PDFPageInterpreter(resource_manager, converter)
    with open(file, 'rb') as fh:
        for page in PDFPage.get_pages(fh,
                                      caching=True,
                                      check_extractable=True):
            page_interpreter.process_page(page)
            print(page)
        text = fake_file_handle.getvalue()

    # close open handles
    converter.close()
    fake_file_handle.close()
    return text
```

Course Recommendation

```
# course recommendations which has data already loaded from Courses.py
def course_recommender(course_list):

    st.subheader("**Courses & Certificates Recommendations 🎓 **")
    c = 0
    rec_course = []
    # slider to choose from range 1-10
    no_of_reco = st.slider('Choose Number of Course Recommendations:', 1, 10, 5)
    random.shuffle(course_list)
```

```
    for c_name, c_link in course_list:
        c += 1
        st.markdown(f"({c}) [{c_name}]({c_link})")
        rec_course.append(c_name)

        if c == no_of_reco:
            break
    return rec_course
```

DB Connection and insertion

```
###### Database Stuffs ######

# sql connector
connection = pymysql.connect(host='localhost',user='root',password='root@MySQL4admin',db='cv')
cursor = connection.cursor()

# inserting prediction and fetched results into database
def
insert_data(sec_token,ip_add,host_name,dev_user,os_name_ver,latlong,city,state,country,act_name,act_mail,act_mob,
name,email,res_score,timestamp,no_of_pages,reco_field,cand_level,skills,recommended_skills,courses,pdf_name):
    DB_table_name = 'user_data'
    insert_sql = "insert into " + DB_table_name + """
    values (0,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"""
    rec_values =
(str(sec_token),str(ip_add),host_name,dev_user,os_name_ver,str(latlong),city,state,country,act_name,act_mail,act_m
ob,name,email,str(res_score),timestamp,str(no_of_pages),reco_field,cand_level,skills,recommended_skills,courses,pd
f_name)
    cursor.execute(insert_sql, rec_values)
    connection.commit()
```

Experience Prediction

```
                    cand_level = ''
                    if resume_data['no_of_pages'] < 1:
                        cand_level = "NA"
                        st.markdown( '''<h4 style='text-align: left; color: #d73b5c;'>You are at Fresher
level!</h4>''',unsafe_allow_html=True)
                    elif 'INTERNSHIP' in resume_text:
                        cand_level = "Intermediate"
                        st.markdown('''<h4 style='text-align: left; color: #1ed760;'>You are at intermediate
level!</h4>''',unsafe_allow_html=True)
                    elif 'INTERNSHIPS' in resume_text:
                        cand_level = "Intermediate"
                        st.markdown('''<h4 style='text-align: left; color: #1ed760;'>You are at intermediate
level!</h4>''',unsafe_allow_html=True)
```

```
                elif 'Internships' in resume_text:
                    cand_level = "Intermediate"
                    st.markdown('''<h4 style='text-align: left; color: #1ed760;'>You are at intermediate
level!</h4>''',unsafe_allow_html=True)
                elif 'EXPERIENCE' in resume_text:
                    cand_level = "Experienced"
                    st.markdown('''<h4 style='text-align: left; color: #fba171;'>You are at experience
level!''',unsafe_allow_html=True)
                elif 'WORK EXPERIENCE' in resume_text:
                    cand_level = "Experienced"
                    st.markdown('''<h4 style='text-align: left; color: #fba171;'>You are at experience
level!''',unsafe_allow_html=True)
                elif 'Work Experience' in resume_text:
                    cand_level = "Experienced"
                    st.markdown('''<h4 style='text-align: left; color: #fba171;'>You are at experience
level!''',unsafe_allow_html=True)
                elif 'Experience' in resume_text:
                    cand_level = "Experienced"
                    st.markdown('''<h4 style='text-align: left; color: #fba171;'>You are at experience
level!''',unsafe_allow_html=True)
                else:
                    cand_level = "Fresher"
                    st.markdown('''<h4 style='text-align: left; color: #fba171;'>You are at Fresher
level!!''',unsafe_allow_html=True)
```

Show predicted skills, interested field with skills recommendations

```
                st.subheader("**Skills Recommendation💡**")
                ## Skill shows
                keywords = st_tags(label='### Your Current Skills',
                text='See our skills recommendation below',value=resume_data['skills'],key = '1 ')
                ##  keywords
                ds_keyword = ['tensorflow','keras','pytorch','machine learning','deep Learning','flask','streamlit']
                recommended_skills = []
                reco_field = '' rec_course = '' ##
                Courses recommendation for i in
                resume_data['skills']:

                    ## Data science recommendation
                    if i.lower() in ds_keyword:
                        print(i.lower())
                        reco_field = 'Data Science'
                        st.success("** Our analysis says you are looking for Data Science Jobs.**")
                        recommended_skills = ['Data Visualization','Predictive Analysis','Statistical']
                        recommended_keywords = st_tags(label='### Recommended skills for you.',
                        text='Recommended skills generated from System',value=recommended_skills,key = '2')
```

```
                    st.markdown('''<h4 style='text-align: left; color: #1ed760;'>Adding this skills to resume
will boost🚀 the chances of getting a Job</h4>''',unsafe_allow_html=True)
                    rec_course = course_recommender(ds_course)
                    break
```

Resume scorer and tips recommender

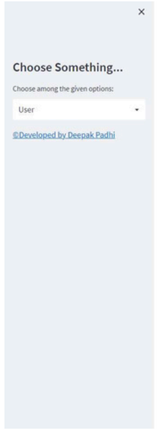
```
                ### Resume writing recommendation
                st.subheader("**Resume Tips & Ideas💡**")
                resume_score = 0
                if 'Objective' or 'Summary' in resume_text:
                    resume_score = resume_score+6
                    st.markdown('''<h5 style='text-align: left; color: #1ed760;'>[+] Awesome! You have added
Objective/Summary</h4>''',unsafe_allow_html=True)
                else:
                    st.markdown('''<h5 style='text-align: left; color: #000000;'>[-] Please add your career
objective, it will give your career intension to the Recruiters.</h4>''',unsafe_allow_html=True)
                st.subheader("**Resume Score📝**")
                st.markdown(
                    """
                    <style>
                        .stProgress > div > div > div > div {
                            background-color: #d73b5c;
                        }
                    </style>""",
                    unsafe_allow_html=True,
                )
                my_bar = st.progress(0)
                score = 0
                for percent_complete in range(resume_score):
                    score +=1
                    time.sleep(0.1)
                    my_bar.progress(percent_complete + 1)
                st.success('** Your Resume Writing Score: ' + str(score)+'**')
                st.warning("** Note: This score is calculated based on the content that you have in your Resume.
**")
```

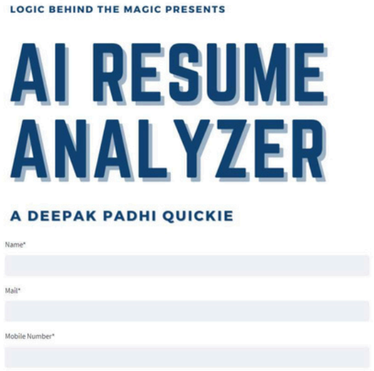
Pie Chart

```
                ## Pie chart for predicted field recommendations
                labels = plot_data.Predicted_Field.unique()
                values = plot_data.Predicted_Field.value_counts()
                st.subheader("**Pie-Chart for Predicted Field Recommendation**")
                fig = px.pie(df, values=values, names=labels, title='Predicted Field according to the Skills',
color_discrete_sequence=px.colors.sequential.Aggrnyl_r)
                st.plotly_chart(fig)
```

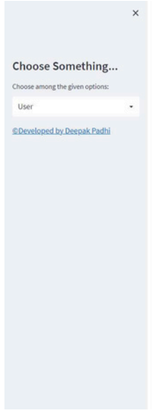
RESULTS AND DISCUSSION

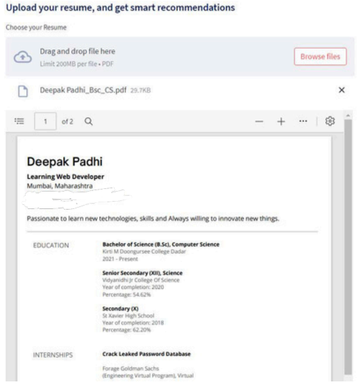
1. Homepage



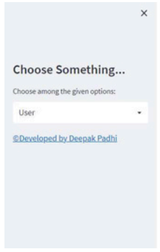


2. Upload and view resume

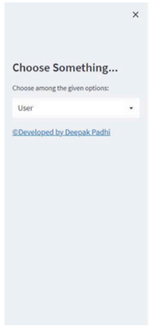


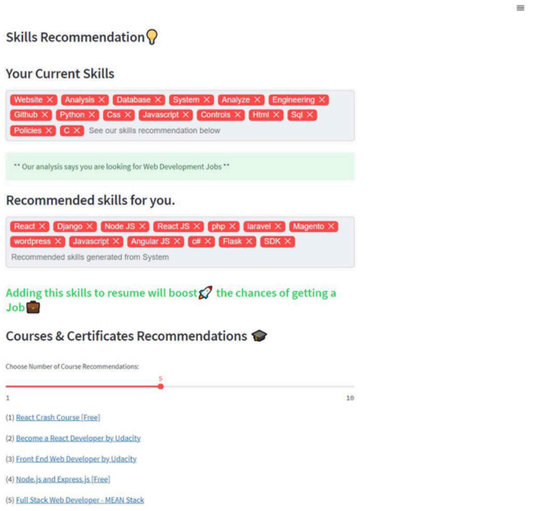


3. Analysis

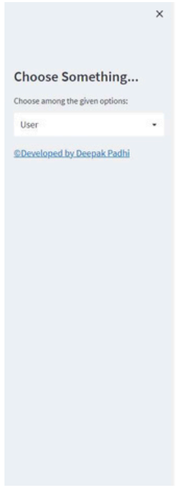


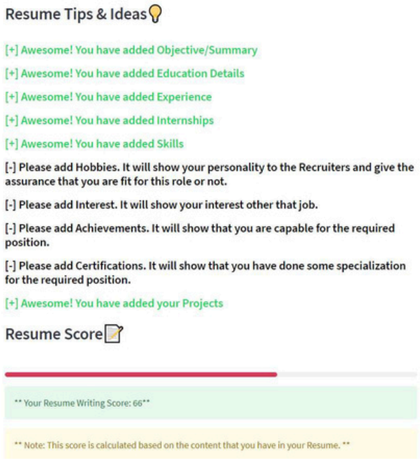
4. Recommendation & Prediction



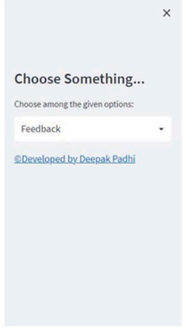


5. Resume tips & ideas with overall Scorer

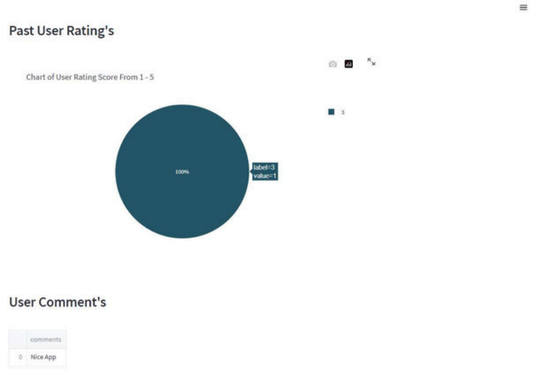




6. Feedback Form

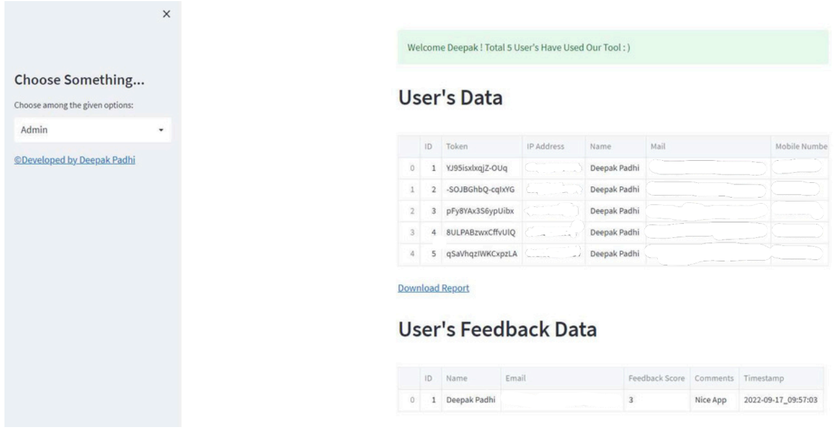


7. Past user ratings and comments

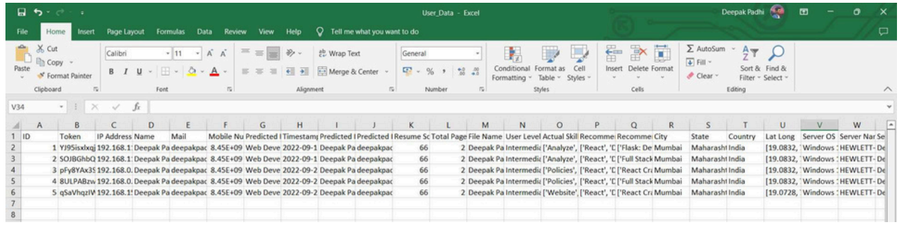


8. Admin Login

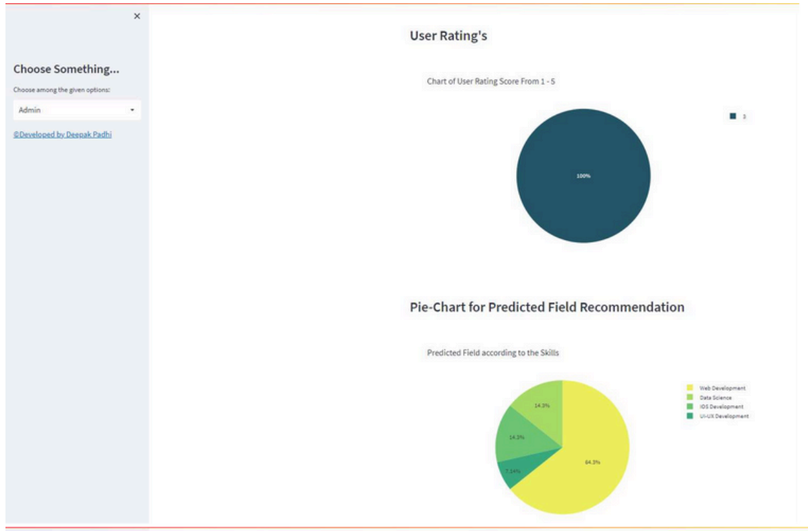
9. Total user’s, user data table, csv file download link, feedback data table



10. Downloaded csv file



11. Analytics Sheet



Both employers and candidates will be appeased by our system. Lots of strain on the head of candidate or employee in Online Recruitment System will have been reduced by this online tool. The system will parse all the resumes and store them in the database. This system is just a small precent from an actual ATS (Application Tracking System). Further it can be well developed to make it like an ATS

Final Conclusion

An applicant cum recruiter-based Quick and easy to use Resume Analyzer. That analyze resume data and extract it into machine-readable output. Helps applicants with recommendations, prediction and analytics. Helps recruiter by automatically store, organize, and analyze resume data to find the best candidate. Can be widely used by any organization to analyze and get insights of a resume

Future Works

Add more fields for other roles, and its recommendations respectively. Ranking out the resume based on score and view individual user details. Decide more accurately and authentically, whether or not to offer candidate a job

Reference

https://docs.streamlit.io/

https://www.udemy.com/course/the-full-stack-data-scientist-bootcamp/

https://www.ijitee.org/wp-content/uploads/papers/v9i7/F4078049620.pdf

https://www.academia.edu/32543544/Resume_Parser_with_Natural_Language_Processing

https://www.rchilli.com/blog/resume-parsing-101/

https://en.wikipedia.org/wiki/R%C3%A9sum%C3%A9_parsing