# MAJOR PROJECT

# ON

**SKILL BASED INTERNSHIP RECOMMENDATION**

Submitted towards the partial fulfillment of the Academic requirement for the award of Degree of

Master of Computer Applications

(Academic Year 2020-21)

SUBMITTED BY:

## Itishree Behera (1805106012)

UNDER THE ESTEEMED GUIDANCE OF:

## Dr. Jibitesh Mishra(HOD,CSA)

**DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(Accredited with grade ’A’ by NAAC and NBA & Affiliated to Biju Patnaik University of Technology)**

**TECHNO CAMPUS, GHATIKIA, BHUBANESWAR, 751029**

**DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**TECHNO CAMPUS, GHATIKIA, BHUBANESWAR, 751029**



# CERTIFICATE

This is to certify that the project entitled, **Skill based internship recommendation system** is a bona fide work done and submitted by **Itishree Behera**, bearing the Registration No– **1805106012**, in partial fulfillment of the requirement for the award “ Master of Computer Applications ” Degree is an authentic record of the work done by her under my supervision and guidance .The matter embodied in the project has not been submitted to any other University for the award of any Degree or Diploma to the best of my knowledge .The work carried out by them during the project period is original and performance during the compilation of project was appreciable.

### INTERNAL GUIDE HOD

**Dr. Jibitesh Mishra Dr. Jibitesh Mishra**

**(Associate Professor) (Associate Professor)**

**DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**TECHNO CAMPUS, GHATIKIA, BHUBANESWAR, 751029**



**DECLARATION**

I hereby declare that the Minor project work entitled **“Skill based internship recommendation system”** submitted to the College of Engineering and Technology, Bhubaneswar is a record of original work done by me under the guidance of Dr. Jibitesh Mishra, Head of the Department of Dept. of Computer Science and Application and this project work is submitted in the partial fulfillment of the award of the degree of **Master of Computer Applications**. The results embodied in this report have not been submitted to any other University or Institutes for the award of any degree.

### Date: 01 /05/2021 Itishree Behera (1805106012)

**DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS COLLEGE OF ENGINEERING AND TECHNOLOGY**

**TECHNO CAMPUS, GHATIKIA, BHUBANESWAR, 751029**



**ACKNOWLEDGEMENT**

What I write or mention in this sheet will hardly be adequate in return for the amount of help and cooperation I have received from all the people who have contributed to make this project a reality. This project owes its existence to the help, support and inspiration of these people.

I would like to express my sincere gratitude to my advisor , **Mr. Jibitesh Mishra,** HOD, Computer Science & Engineering Department, whose knowledge and guidance has motivated me to achieve goals I never thought possible. He has consistently been a source of motivation, encouragement, and inspiration. The time I have spent working under his supervision has truly been a pleasure.

I would also like to convey our deep regards to all other faculty members of Dept. of CSA, who have bestowed their great effort and guidance at appropriate times without which it would have been very difficult on our part to finish this work. Finally, I would also like to thank my friends for their advice and pointing out my mistakes.

**Date: 01/05/2021** **Itishree Behera (1805106012)**

**Abstract**

There are many ways one can get an internship. Sometimes companies come to college to hire interns, students apply directly for internships on the company's portal or one can seek a referral to get the internship. To deal with the information overload for students during their transition into work, an internship recommendation system can be very valuable. While it’s a difficult job for the company to verify the resume of each and every applied students and filter out the best matched profile.

Our system can automate the task of matching the student’s profile with the company’s requirements based on the skill set. We have proposed a student skill based internship recommendation system where students are recommended a list of potential internships based on their skill set. When a new internship is posted by the company, based on the company requirement the system will recommend the internship to the students having those required skills and push notification through email. When one student apply the particular internship, the system will show the matching skill percentage to the company. When one company accepts the student’s application, the system will generate one certificate of acceptance and send the certificate to the particular student.

**Table of Content**

|  |  |
| --- | --- |
| **Description** | **Page no.** |
| Introduction -------------------------------------------------------------------- | 1 |
| Software and hardware requirements --------------------------------------- | 3 |
| Literature Survey -----------------------------------------------------------------  Software requirement analysis--------------------------------------------------  Software design --------------------------------------------------------------- | 5  7  9 |
| Coding -------------------------------------------------------------------------- | 16 |
| Output screen ----------------------------------------------------------------------  Bibliography -------------------------------------------------------------------- | 36  43 |

**Chapter 1. INTRODUCTION**

**1. Introduction**

When a student wants an internship, it is often difficult since there is too much internship available with different skill sets. To select an appropriate internship and then submit a resume is a tedious job. It is particularly difficult for students since they normally are unfamiliar with the job market . To deal with the information overload for students during their transition into work, an internship recommendation system can be very valuable. While its a difficult job for the company to verify the resume of each and every applied students and filter out the best matched profile. This system can automate the task of matching the student’s profile with the company’s requirements based on the skill set.

* We have proposed a student skill based internship recommendation system where students are recommended a list of potential internships based on their skill set.
* When a new internship is posted by the company, based on the company requirement the system will recommend the internship to the students having those required skills and push notification through email. In this system we are using jaccard similarity to create the ranking framework.
* When one student apply the particular internship ,the system will show the matching skill percentage to the company.
* When one company accepts the student’s application , the system will generate one certificate of acceptance and send the certificate to the particular student.

**Chapter 2. Software and Hardware requirements**

# 2. Software and Hardware requirements

1. **Software Requirements :**
   1. **. Front‐end**
      1. HTML
      2. CSS
      3. Javascript
      4. Boostrap
   2. **Back-end**
      1. Django
      2. Jacard similarity index
   3. **IDE**
      1. PYcharm
   4. **Database**
      1. SQLite
2. **Hardware Requirements :**

**2.1. Application Requirement**

2.1.1. 4GB RAM

2.1.2. 1.2 GHZ Processor

**2.2. Development Environment Requirement**

2.2.1. Intel i3 5th Generation Processor

2.2.2. Windows 10 / UBUNTU 19.0.4

**Chapter 3. Literature survey**

**Literature survey**

There are endless algorithms to help a seeker find the right internship, some are the traditional algorithms while some are newly found and there are a large number of hybrid algorithms which are a combination of many algorithms. All these algorithms have tried to recommend the best internship/job to a student.

Similar to my task Vinay Desai, Dheeraj Bahl, Shreekumar Vibhandik,Isra Fatma have published a paper “Implementation of an Automated Job Recommendation System Based on Candidate Profiles” where they contrast user-based and item-based collaborative filtering algorithm to choose a better performed one. they also took background information including students’ resumes and details of recruiting information into consideration, bring weights of co-apply users (the users who had applied the candidate jobs) and weights of student used liked jobs into their commendation algorithm. [1]

Jorge Valverde-Rebaza Ricardo Puma Paul Bustios Nathalia C. Silva used two methods used in their experiments: Term Frequency-Inverse Document Frequency (TF-IDF) and Word2vec. . In the case of TF-IDF representation, we use the cosine distance while for word embeddings, they used the relatively new Word Mover’s Distance (WMD) [Kus15].[2]

W. Hong et al. developed iHR an online job recommendation system that classifies users into groups by using historical behaviors of users and individual information and then uses the appropriate recommendation approach for each group of users. This approach is suitable for the cases in which different users may have different attributes and a single recommendation approach may not be appropriate for all users[3]

Mamadou et al. presented an online social network-based recommender system that extracts users’ interests for jobs and then make recommendations according to user’s interest.[4]

In addition to these many e-commerce websites, uses collaborative filtering algorithm without considering user’s resume and item’s properties[5]

# Chapter 4. Software Requirement Analysis

**4. Software Requirement Analysis**

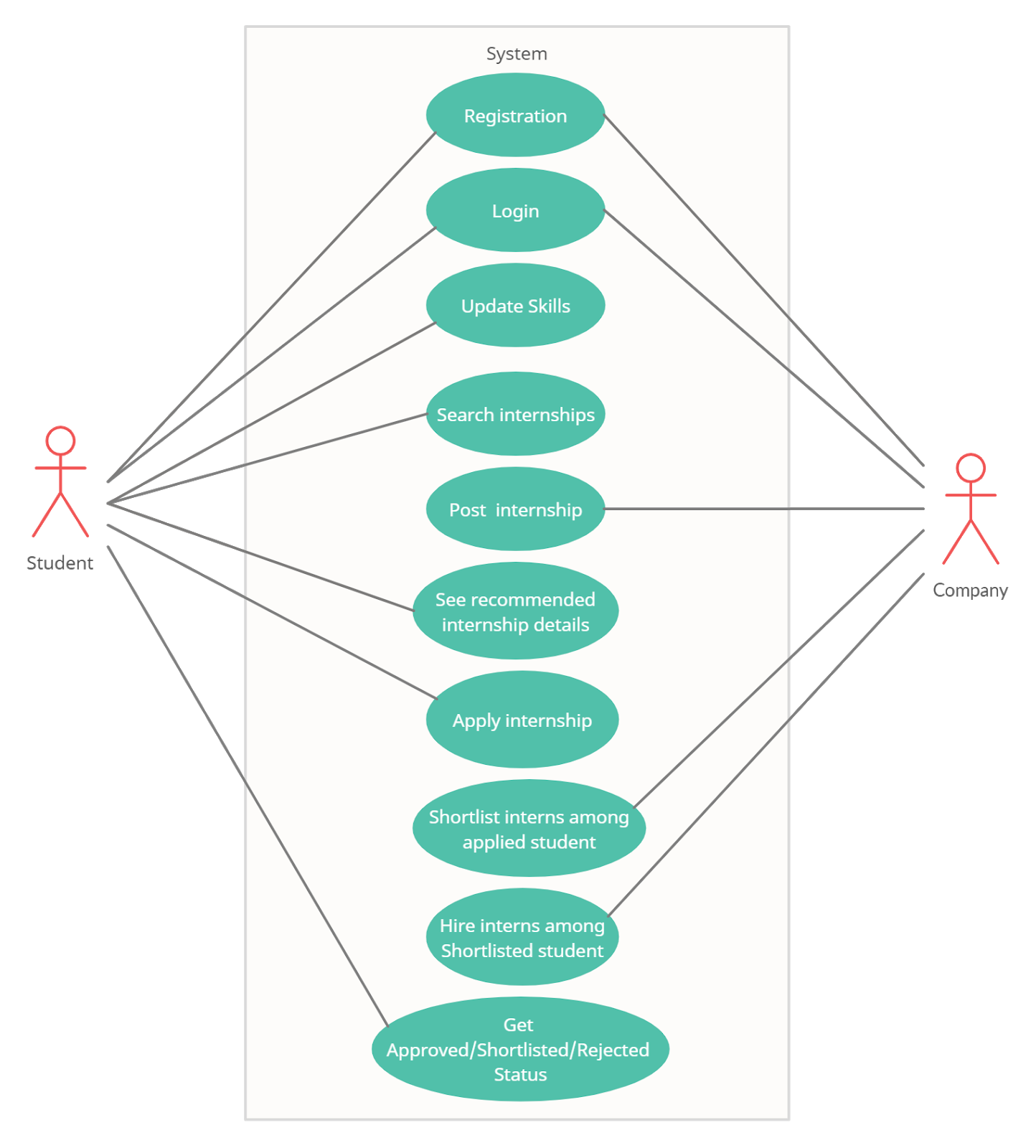
1. **Problem**

* Since there are many ways sometimes it becomes unmanageable if one tries to grab internship opportunity from all the available sources.
* Finding internship with matching skill set requires lots of searching
* It’s a difficult job for the company to verify the resume of each and every applied student and filter out the best matched profile.
* To recommend a job to a student, a dataset is required which use the previous data of that student and recommend accordingly, but here in our case we don’t use dataset rather we use the similarity index which find the instant matched percentage and push notification.

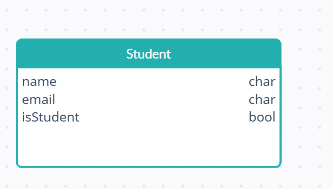
1. **Modules and their functionalities**
2. Match the skills set of student and company and find out the percentage similarity
3. Send an recommendation mail to the student
4. Profile and resume match
5. Auto generate certificate
6. Send an internship accepted mail when one company accepts the application.

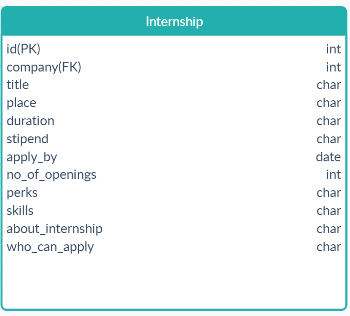
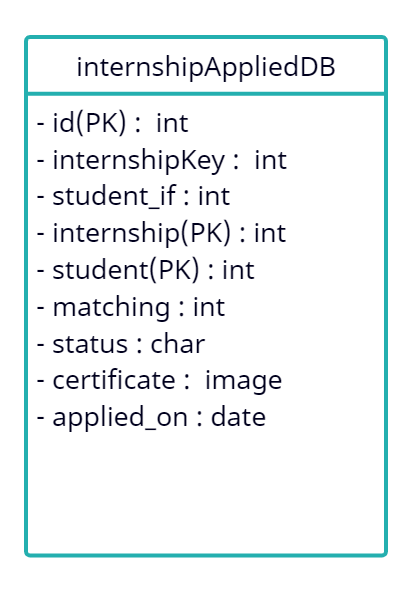
**Chapter 5. Software design**

**Use case diagram**

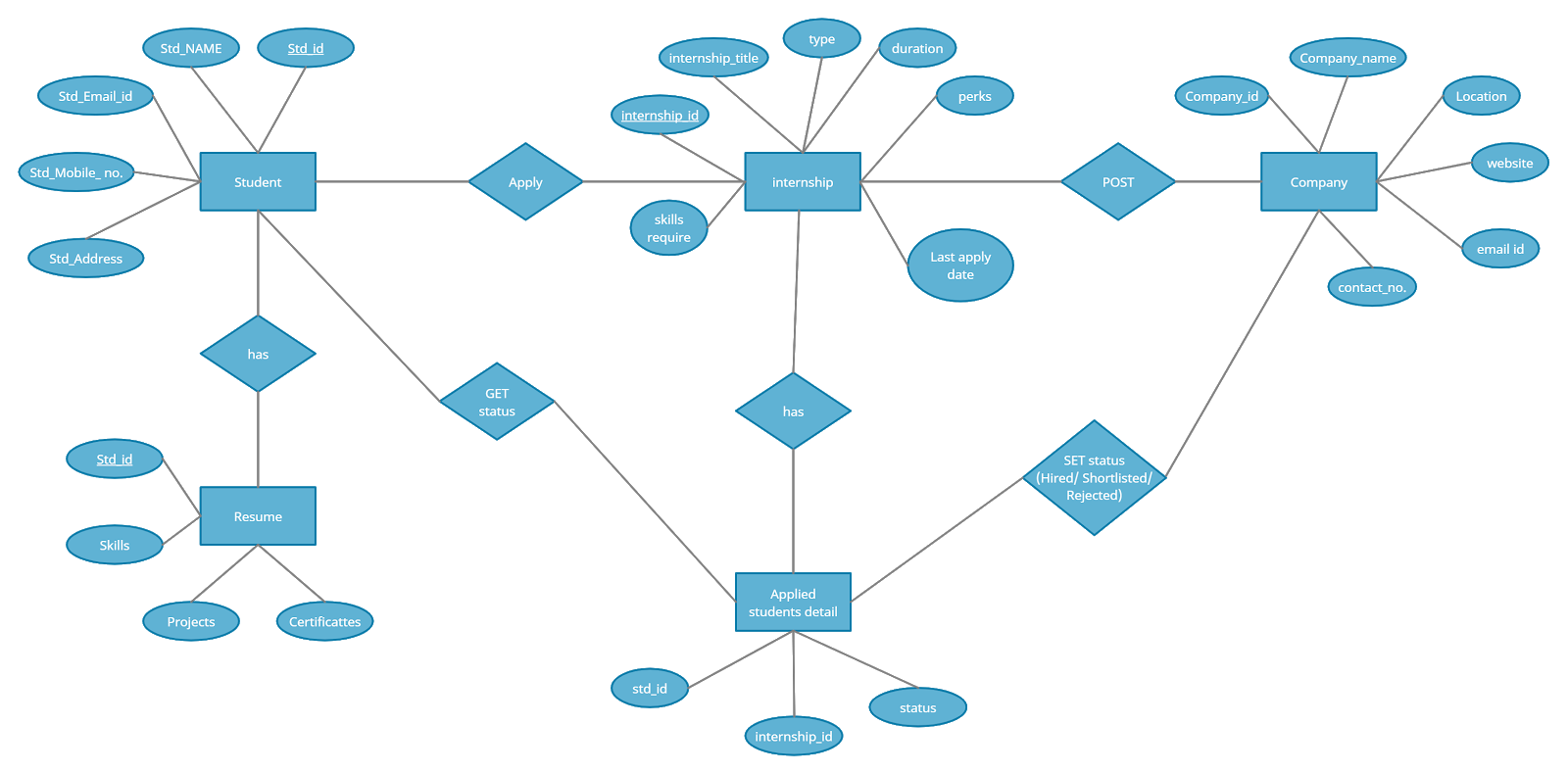


**Class diagram:**

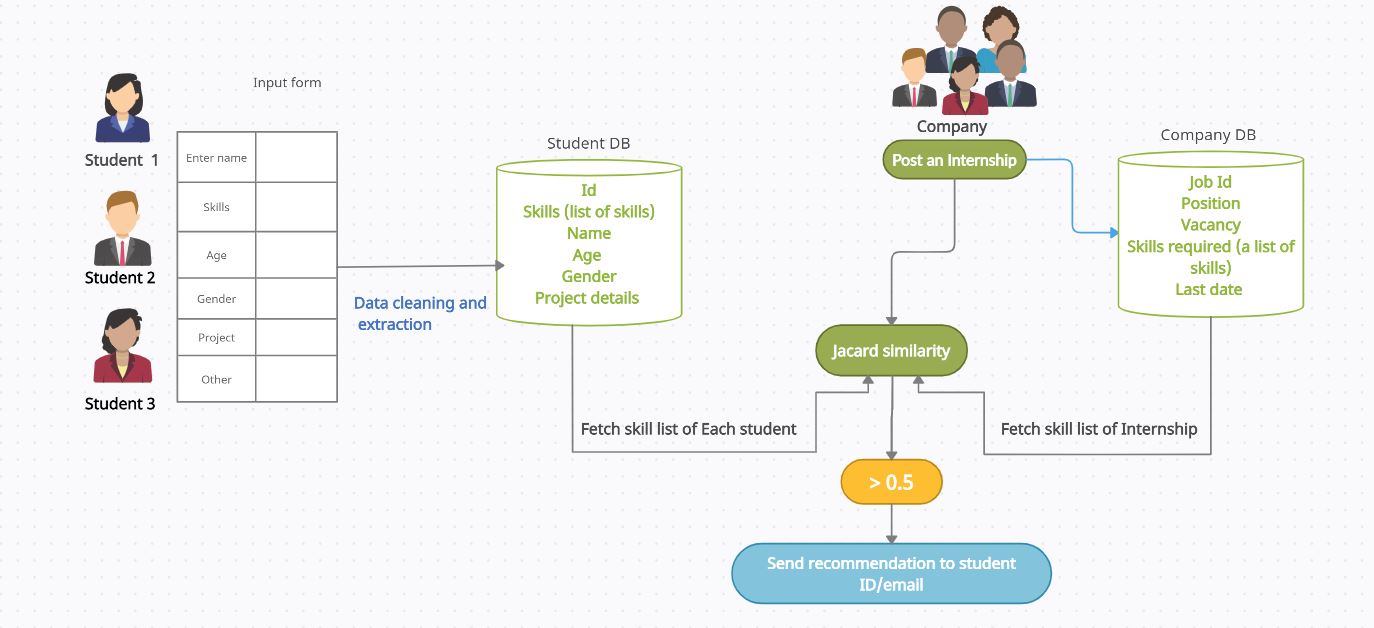
** **

** **

**ER diagram**



**System architecture:**

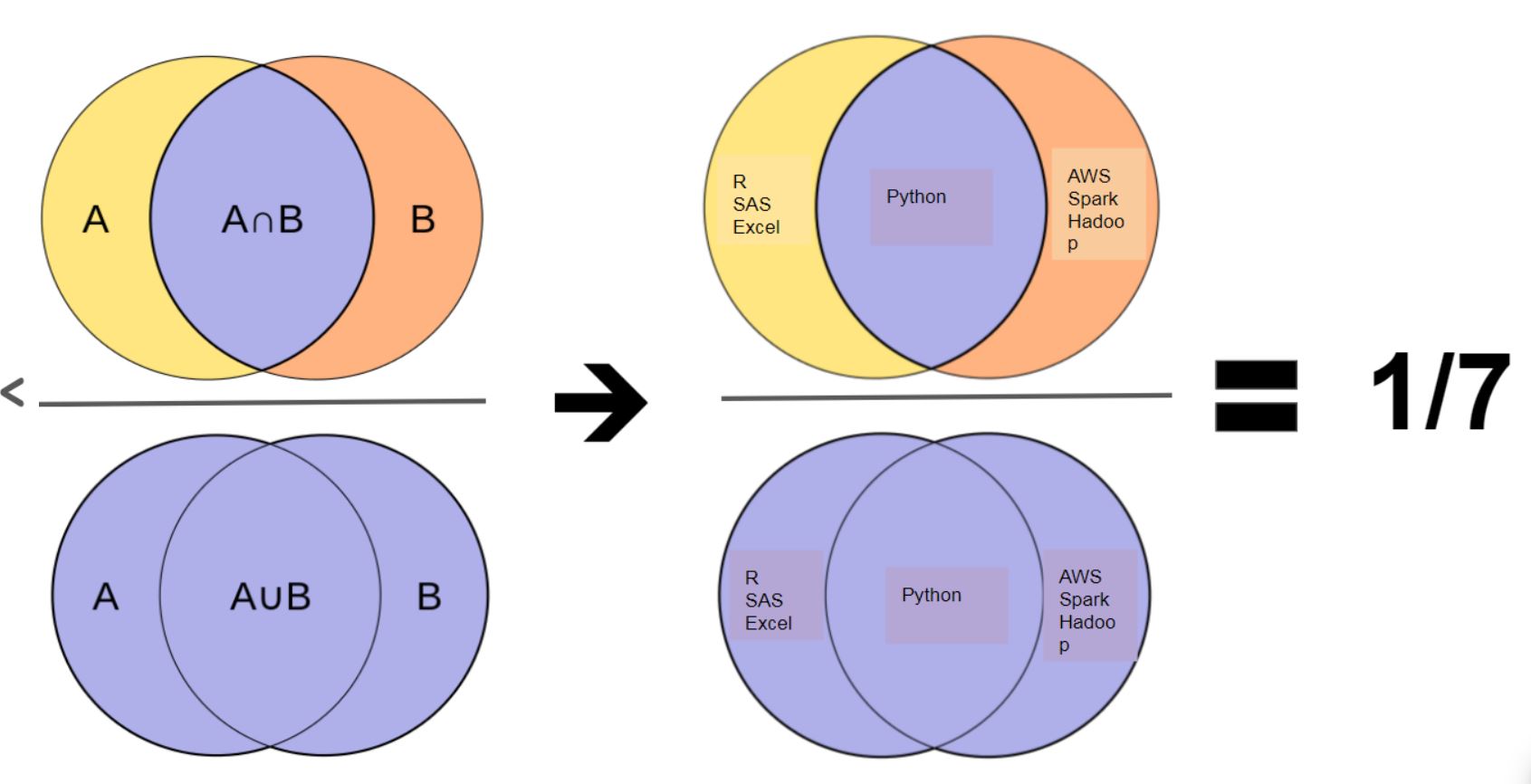
****

**Jaccard similarity:**

The Jaccard index is a statistic used for comparing the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets.

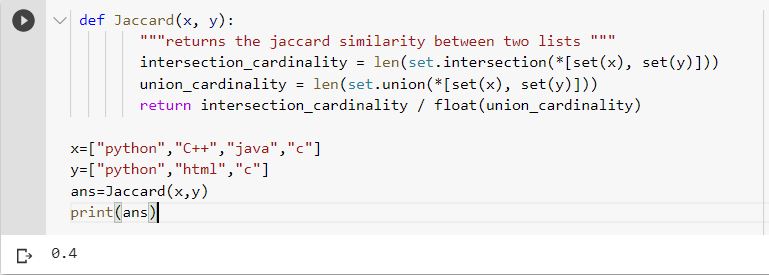
In this scenario,

A represents student skill-sets and B represents internship requirements skill list.

****

X: list of skills a student have

Y: List of skills required for a particular internship by the company

****

**Chapter 6.Coding**

**Source code**

**Student/models.py**

from django.db import models

from django.contrib.auth.models import User

from jsonfield import JSONField

# Create your models here.

class Student(models.Model):

objects = models.Manager()

user=models.OneToOneField('auth.user',default="", on\_delete=models.CASCADE)

name = models.CharField(default="", max\_length=128)

email = models.EmailField()

isStudent = models.BooleanField(default=True)

def \_\_str\_\_(self):

return self.name

class Resume(models.Model):

objects = models.Manager()

student = models.OneToOneField(Student, on\_delete=models.CASCADE)

address = models.CharField(blank=True, null=True, max\_length=128)

mob = models.CharField(blank=True, null=True, max\_length=10)

skills = models.TextField(blank=True, null=True)

pic = models.ImageField(upload\_to="student/", blank=True, null=True)

college = models.CharField(blank=True, null=True, max\_length=128)

grad\_year = models.CharField(blank=True, null=True, max\_length=10)

cgpa = models.CharField(blank=True, null=True, max\_length=5)

resume\_file = models.FileField(upload\_to="student/resume/", blank=True, null=True)

def \_\_str\_\_(self):

return self.student.name

class Meta:

db\_table = ''

managed = True

verbose\_name = 'Resume'

verbose\_name\_plural = 'Resumes'

class Certificate(models.Model):

objects = models.Manager()

id = models.AutoField(primary\_key=True)

std\_id = models.IntegerField()

title = models.CharField(max\_length=1000, blank=True, null=True)

company\_name = models.CharField(max\_length=1000, blank=True, null=True)

complition\_year = models.IntegerField()

credential = models.CharField(max\_length=100, blank=True, null=True)

certificate\_file = models.FileField(upload\_to="student/certificate/", blank=True, null=True)

def \_\_str\_\_(self):

return self.title

**company/models.py**

from django.db import models

from django.contrib.auth.models import User

# Create your models here.

class Company(models.Model):

objects = models.Manager()

user=models.OneToOneField('auth.user',default="", on\_delete=models.CASCADE)

name = models.CharField(default="", max\_length=128)

email = models.EmailField()

isCompany = models.BooleanField(default=True)

def \_\_str\_\_(self):

return self.name

class Meta:

managed = True

verbose\_name = 'Company'

verbose\_name\_plural = 'Companies'

class Profile(models.Model):

objects = models.Manager()

company = models.OneToOneField(Company, on\_delete=models.CASCADE)

pic = models.ImageField(upload\_to="company/", blank=True, null=True)

mob = models.CharField(blank=True, null=True, max\_length=10)

address = models.CharField(blank=True, null=True, max\_length=128)

website = models.URLField(blank=True, null=True, max\_length = 200)

no\_of\_employees = models.IntegerField(default=0, null=True)

internship\_post = models.IntegerField(default=0, null=True)

interns\_hired = models.IntegerField(default=0, null=True)

facebook\_link = models.URLField(blank=True, null=True, max\_length = 200)

twitter\_link = models.URLField(blank=True, null=True, max\_length = 200)

linkedin\_link = models.URLField(blank=True, null=True, max\_length = 200)

youtube\_link = models.URLField(blank=True, null=True, max\_length = 200)

about = models.TextField(blank=True, null=True)

def \_\_str\_\_(self):

return self.company.name

class Meta:

managed = True

verbose\_name = 'Profile'

verbose\_name\_plural = 'Profiles'

class Internship(models.Model):

objects = models.Manager()

company = models.ForeignKey( 'Company' , on\_delete=models.CASCADE)

title = models.CharField(max\_length=128)

place = models.CharField(max\_length=128)

duration = models.CharField(max\_length=128)

stipend = models.CharField(max\_length=128)

apply\_by = models.DateField()

no\_of\_openings = models.IntegerField()

perks = models.CharField(max\_length=128)

skills = models.CharField(max\_length=128)

about\_internship = models.TextField()

who\_can\_apply = models.TextField()

def \_\_str\_\_(self):

return str(self.title)

class Meta:

db\_table = ''

managed = True

verbose\_name = 'Internship'

verbose\_name\_plural = 'Internships'

class InternshipAppliedDB(models.Model):

id = models.AutoField(primary\_key=True)

internship\_id = models.IntegerField(null=False)

student\_id = models.IntegerField(null=False)

student\_name = models.CharField(default="", max\_length=128)

student\_email = models.EmailField(default=True, null=True)

student\_mob = models.CharField(blank=True, null=True, max\_length=10)

matching = models.DecimalField(default=True, null=True, max\_digits=5, decimal\_places=2)

status = models.CharField(max\_length=30, default="pending")

certificate = models.ImageField(upload\_to="student/certificate/", blank=True, null=True)

def \_\_str\_\_(self):

return str(self.internship\_id)

**email\_template.html**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

<title>Demystifying Email Design</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<style type="text/css">

a[x-apple-data-detectors] {color: inherit !important;}

</style>

</head>

<body style="margin: 0; padding: 0;">

<table role="presentation" border="0" cellpadding="0" cellspacing="0" width="100%">

<tr>

<td style="padding: 20px 0 30px 0;">

<table align="center" border="0" cellpadding="0" cellspacing="0" width="600" style="border-collapse: collapse; border: 1px solid #cccccc;">

<tr>

<td align="center" bgcolor="#70bbd9" >

<img src='https://www.linkpicture.com/q/Screenshot-16\_2.png' type='image' alt="Creating Email Magic." width="100%" style="display: block;" />

</td>

</tr>

<tr style="border-bottom: 1px solid #ddd;">

<td style="padding: 10px;">

<p>Hey, <strong>{{name}}</strong></p>

<p>Here is a new intership opening that match your profile.

Just click on <a href="{{ link }}" style="text-decoration: none;">view details</a> to read about the internship and apply.</p>

</td>

</tr>

<tr>

<td bgcolor="#fff" style="padding: 30px 30px; border-bottom: 1px solid #ddd;">

<table border="0" cellpadding="0" cellspacing="0" width="100%" style="border-collapse: collapse;">

<tr>

<td >

<table border="0" cellpadding="0" cellspacing="0" style="border-collapse: collapse;">

<tr>

<td style="text-align: center;">

<h2>{{cmp\_name}}</h2>

<img src='https://www.linkpicture.com/q/1200px-Wipro\_logo.svg.png' type='image' alt="Company Name" width="100px" height="100px" style="margin: auto;" border="0" />

</td>

</tr>

</table>

</td>

<td style=" font-family: Arial, sans-serif; font-size: 14px; text-align: center;">

<h3 style="margin-bottom: 0px;">{{pos}} </h3>

<p style="color: #555; font-size: 13px; margin-bottom: 20px; margin-top: 5px;">{{place}}</p>

<a href="{{ link }}" style="padding: 10px; text-decoration:none; background-color:#0058a6; color: #fff; font-weight: bold;">View & Apply</a>

</td>

</tr>

</table>

</td>

</tr>

<tr>

<td style="text-align: center;">

<p>

Simply intern team. <br>

Happy learning!!!

</p>

</td>

</tr>

<tr>

<td bgcolor="#ee4c50" style="padding: 30px 30px;">

<table border="0" cellpadding="0" cellspacing="0" width="100%" style="border-collapse: collapse;">

<tr>

<td style="color: #ffffff; font-family: Arial, sans-serif; font-size: 14px;">

<p style="margin: 0;">&reg; SimplyIntern<br/>

<a href="#" style="color: #ffffff;">Unsubscribe</a> to this newsletter instantly</p>

</td>

<td align="right">

<table border="0" cellpadding="0" cellspacing="0" style="border-collapse: collapse;">

<tr>

<td>

<a href="http://www.twitter.com/">

<img src="https://assets.codepen.io/210284/tw.gif" alt="Twitter." width="38" height="38" style="display: block;" border="0" />

</a>

</td>

<td style="font-size: 0; line-height: 0;" width="20">&nbsp;</td>

<td>

<a href="http://www.twitter.com/">

<img src="https://assets.codepen.io/210284/fb.gif" alt="Facebook." width="38" height="38" style="display: block;" border="0" />

</a>

</td>

</tr>

</table>

</td>

</tr>

</table>

</td>

</tr>

</table>

</td>

</tr>

</table>

</body>

</html>

**company/views.py**

from django.shortcuts import render, redirect, reverse

from django.contrib.auth import login, authenticate

from django.contrib.auth.decorators import login\_required

from django.contrib.auth.models import User, auth

from django.http import HttpResponse , JsonResponse

from .models import Company, Internship, Profile, InternshipAppliedDB

from student.models import Student,Resume

from django.contrib import messages

from django.core.mail import EmailMessage

from email.mime.image import MIMEImage

from functools import lru\_cache

from django.contrib.staticfiles import finders

from django.conf import settings

from django.core.mail import EmailMultiAlternatives

from PIL import Image, ImageDraw, ImageFont

import pandas as pd

import os

from django.core.mail import EmailMultiAlternatives

from django.core.mail import send\_mail

def signin(request):

password = request.POST.get('password')

username = request.POST.get('username')

if (request.POST.get('formtype') =='signupform'):

username = request.POST.get('email')

print(username, password)

user = auth.authenticate(username=username,password=password)

if user is not None :

try:

if (user.company.isCompany == True ) :

auth.login(request,user)

print(user.company.name, 'logged in')

return True

else:

print("not company")

return False

except :

print("company error")

return False

else:

print("not user")

return False

def auth\_company(request):

if request.method == 'POST':

if (request.POST.get('formtype') =='signupform'):

username = request.POST.get('email')

name = request.POST.get('name')

email = request.POST.get('email')

password = request.POST.get('password')

password1 = request.POST.get('confpassword')

if password == password1:

if User.objects.filter(username=username).exists():

messages.info(request, 'User name already exists')

return redirect(auth\_company)

# return HttpResponse('id exists')

else:

user = User.objects.create\_user(username=username,first\_name=name, password=password, email=email)

user.save()

newCompany = Company(user=user, name=name, email=email)

newCompany.save()

newProfile = Profile(company=newCompany)

newProfile.save()

# messages.info(request, 'User Created Successfully')

signin(request)

messages.info(request, 'Sucessfully Registered and signed in.')

return redirect(home)

# return redirect('login')

elif (request.POST.get('formtype')=='signinform'):

if signin(request):

messages.info(request, "Signed in successfully")

return redirect(home)

else:

messages.info(request, "invlid credentials....")

return redirect(auth\_company)

else:

messages.info(request, "invlid credentials here....")

return redirect(auth\_company)

else:

return render(request,'Company.html')

def home(request):

if request.user.is\_authenticated:

try:

if (request.user.company.isCompany == True):

comp = Company.objects.get(email=request.user.company.email)

print(comp)

posts = Internship.objects.filter(company=comp).order\_by('-id')

print(posts)

return render(request, 'CompanyHomePage2.html', {'posts':posts, 'company':comp})

else:

messages.info(request, 'Please sign in as a company')

return redirect(auth\_company)

except:

print(request.user.company.isCompany, request.user.company)

messages.info(request, 'Some error occured. Please signin again')

return redirect(auth\_company)

else:

messages.info(request, 'Please signin to proceed')

return redirect(auth\_company)

def new\_post(request):

std =Student.objects.order\_by('id')

if request.method == 'POST':

if request.user.is\_authenticated:

try:

if (request.user.company.isCompany == True):

title = request.POST.get('title')

place = request.POST.get('place')

duration = '2 Months'

stipend = request.POST.get('stipend')

apply\_by = request.POST.get('apply\_by')

no\_of\_openings = request.POST.get('no\_of\_openings')

perks = request.POST.get('perks')

skills = request.POST.get('skills')

about\_internship = request.POST.get('about\_internship')

who\_can\_apply = request.POST.get('who\_can\_apply')

comp = Company.objects.get(email=request.user.company.email)

newPost = Internship(company=comp, title=title, place=place, duration=duration, stipend=stipend, apply\_by=apply\_by, no\_of\_openings=no\_of\_openings, perks=perks, skills=skills, about\_internship=about\_internship, who\_can\_apply=who\_can\_apply)

# newPost = Internship(company=request.user, title='title', place='place', duration='duration', stipend='stipend', apply\_by='apply\_by', no\_of\_openings='no\_of\_openings', perks='perks', skills='skills', about\_internship='about\_internship', who\_can\_apply='who\_can\_apply')

newPost.save()

print("saved in database")

recommend(skills, std,newPost.id,comp,place,title)

return redirect(home)

except:

print('4 ', request.user, request.user.company.isCompany, request.user.is\_authenticated)

return redirect(auth\_company)

else:

print('4 ', request.user, request.user.company.isCompany, request.user.is\_authenticated)

return redirect(auth\_company)

return render(request,'CompanyInternshipForm.html')

@login\_required

def company\_profile(request):

cmp = request.user.company

profile = cmp.profile

return render(request, 'CompanyProfile.html', {'profile':profile, 'company':cmp})

@login\_required

def company\_profile\_edit(request):

cmp = request.user.company

profile = cmp.profile

if request.method == 'POST':

print(cmp)

email = request.POST.get('email')

if email != cmp.email:

if User.objects.filter(username=email).exists():

messages.info(request, "This email is already in use")

else:

cmp.email = email

request.user.email = email

request.user.username = email

cmp.name = request.POST.get('name')

fullname = cmp.name.split()

request.user.first\_name = fullname[0]

request.user.last\_name = fullname[-1]

cmp.save()

request.user.save()

if 'image' in request.FILES:

cmp.profile.pic = request.FILES['image']

cmp.profile.mob = request.POST.get('mob')

cmp.profile.address = request.POST.get('address')

cmp.profile.website = request.POST.get('website')

cmp.profile.no\_of\_employees = request.POST.get('no\_of\_employees')

cmp.profile.internship\_post= request.POST.get('internship\_post')

cmp.profile.interns\_hired = request.POST.get('interns\_hired')

cmp.profile.facebook\_link = request.POST.get('facebook\_link')

cmp.profile.twitter\_link = request.POST.get('twitter\_link')

cmp.profile.linkedin\_link = request.POST.get('linkedin\_link')

cmp.profile.youtube\_link = request.POST.get('youtube\_link')

cmp.profile.about = request.POST.get('about')

cmp.profile.save()

return redirect('company-profile')

return render(request, 'CompanyProfileEdit.html', {'profile':profile, 'company':cmp})

def post\_detail(request, post\_id):

comp = Company.objects.get(email=request.user.company.email)

post = Internship.objects.get(id=post\_id)

applied = InternshipAppliedDB.objects.filter(internship\_id=post\_id)

return render(request, 'CompanyInternshipDetails.html',{'post': post, 'company': comp, 'applied':applied})

from django.core.mail import EmailMessage

@lru\_cache()

def logo\_data():

with open(finders.find('student//certificates//ItishreeBeheraSoftdev.png'), 'rb') as f:

logo\_data = f.read()

logo = MIMEImage(logo\_data)

logo.add\_header('Content-ID', '<logo>')

return logo

def acceptStd(request, post\_id, a\_id):

internship = InternshipAppliedDB.objects.get(id = a\_id)

internship.status = "Accept"

internship.save()

#send mail to the accpeted students

post=Internship.objects.get(id=post\_id)

#got the company name and position

print(post.title)

print(post.company)

name = internship.student\_name

font = ImageFont.truetype('arial.ttf', 60)

img = Image.open('company/static/images/certificate.jpeg')

draw = ImageDraw.Draw(img)

draw.text(xy=(403, 421), text='{}'.format(name), fill=(0, 0, 0), font=font)

draw.text(xy=(785, 500), text='{}'.format(post.title), fill=(0, 0, 0), font=ImageFont.truetype('arial.ttf', 25))

draw.text(xy=(154, 649), text='{}'.format(post.company), fill=(0, 0, 0), font=font)

img\_name =name+post.title

img.save('media\student\certificates\{}.png'.format(img\_name))

path=('student\certificates\{}.png'.format(img\_name))

internship.certificate = path

internship.save()

print(internship.certificate)

#send confirmation mail and certificate

to = internship.student\_email

send\_mail(

"congrats",

'The internship you have applied have accepted your application',

'simplyintern08@gmail.com',

[to],

fail\_silently=False,

)

return redirect('post-detail', post\_id = post\_id)

def rejectStd(request, post\_id, a\_id):

internship = InternshipAppliedDB.objects.get(id=a\_id)

internship.status = "Reject"

internship.save()

return redirect('post-detail', post\_id = post\_id)

def Jaccard(x, y):

"""returns the jaccard similarity between two lists """

intersection\_cardinality = len(set.intersection(\*[set(x), set(y)]))

union\_cardinality = len(set.union(\*[set(x), set(y)]))

return intersection\_cardinality / float(union\_cardinality)

from django.core.mail import EmailMessage, send\_mail

from django.conf import settings

from django.template.loader import render\_to\_string

def recommend(cmp\_skills,std,id,cmp\_name,place,pos):

print("recommendation!!!!!----------------------")

print("company skills : " ,cmp\_skills)

x = list(cmp\_skills)

for i in std:

print(i.name)

res = i.resume

print("student skills:",res.skills)

y = list(res.skills)

ans = Jaccard(x, y)

print("ans:",ans)

url\_str="http://127.0.0.1:8000/internships/detail/"

url\_str=url\_str+str(id)

if ans >0.5:

template\_txt = render\_to\_string('email\_template.html', {'name':i.name,'link':url\_str,'cmp\_name':cmp\_name,'pos':pos,'place':place})

template\_html = render\_to\_string('email\_template.html', {'name': i.name, 'link': url\_str,'cmp\_name':cmp\_name,'pos':pos,'place':place})

email=send\_mail(

'Internship recommendation',

template\_txt,

settings.EMAIL\_HOST\_USER,

[i.email],

html\_message = template\_html,

)

# email.fail\_slently=False

# email.send()

# -------------------------------

print("successfully emailed")

else:

pass

def std\_profile(request, std\_id):

std = Student.objects.get(id = std\_id)

return render(request, 'StdProfile.html', {'std': std})

**student/views.py**

from django.shortcuts import render, redirect, reverse

from django.contrib.auth import login, authenticate

from django.contrib.auth.decorators import login\_required

from django.contrib.auth.models import User, auth

from django.http import HttpResponse , JsonResponse

from .models import Student, Resume, Certificate

from company.models import Company, Internship, InternshipAppliedDB

from django.contrib import messages

def home(request):

return render(request, 'index.html')

def internships(request):

if request.user.is\_authenticated:

try:

if (request.user.student.isStudent == True):

posts = Internship.objects.order\_by('-id')

std = Student.objects.get(email=request.user.student.email)

return render(request, 'StudentHomePage.html', {'posts':posts, 'student':std})

else:

messages.info(request, 'Please sign in as a student')

return redirect(auth\_student)

except:

messages.info(request, 'Some error occured. Please signin again')

return redirect(auth\_student)

else:

messages.info(request, 'Please signin to proceed')

return redirect(auth\_student)

def signin(request):

password = request.POST.get('password')

username = request.POST.get('username')

if (request.POST.get('formtype') =='signupform'):

username = request.POST.get('email')

user = auth.authenticate(username=username,password=password)

if user is not None :

try:

if (user.student.isStudent == True ) :

auth.login(request,user)

print(user.student.name, 'logged in')

return True

else:

return False

except :

return False

else:

return False

def auth\_student(request):

if request.method == 'POST':

if (request.POST.get('formtype') =='signupform'):

username = request.POST.get('email')

name = request.POST.get('name')

email = request.POST.get('email')

password = request.POST.get('password')

password1 = request.POST.get('confpassword')

fullname = name.split()

firstname = fullname[0]

lastname = fullname[-1]

if password == password1:

if User.objects.filter(username=username).exists():

messages.info(request, 'User name already exists')

return redirect(auth\_student)

# return HttpResponse('id exists')

else:

user = User.objects.create\_user(username=username,first\_name=firstname, last\_name=lastname, password=password, email=email)

user.save()

newStudent = Student(user=user, name=name, email=email)

newStudent.save()

stdResume = Resume(student=newStudent)

stdResume.save()

# messages.info(request, 'User Created Successfully')

signin(request)

messages.info(request, 'Sucessfully Registered and signed in.')

return redirect('internships')

# return redirect('login')

elif (request.POST.get('formtype')=='signinform'):

if signin(request):

messages.info(request, "Signed in successfully")

return redirect('internships')

else:

messages.info(request, "invlid credentials")

return redirect(auth\_student)

else:

return render(request,'Student.html')

@login\_required

def profile(request):

std = request.user.student

resume = std.resume

certificate = Certificate.objects.filter(std\_id = request.user.pk).order\_by('-id')

return render(request, 'StudentProfile.html', {'resume':resume, 'student':std, 'certificate': certificate})

@login\_required

def profileEdit(request):

std = request.user.student

resume = std.resume

if request.method == 'POST':

email = request.POST.get('email')

if email != std.email:

if User.objects.filter(username=email).exists():

messages.info(request, "This email is already in use")

else:

std.email = email

request.user.email = email

request.user.username = email

std.name = request.POST.get('name')

fullname = std.name.split()

request.user.first\_name = fullname[0]

request.user.last\_name = fullname[-1]

std.save()

request.user.save()

if 'image' in request.FILES:

std.resume.pic = request.FILES['image']

if 'resume\_file' in request.FILES:

std.resume.resume\_file = request.FILES['resume\_file']

std.resume.mob = request.POST.get('mob')

std.resume.address = request.POST.get('address')

std.resume.skills = request.POST.get('skills')

std.resume.college = request.POST.get('college')

std.resume.degree= request.POST.get('degree')

std.resume.grad\_year = request.POST.get('grad\_year')

std.resume.cgpa = request.POST.get('cgpa')

std.resume.save()

return redirect('profile')

print(std.name)

return render(request, 'StudentProfileEdit.html', {'resume':resume, 'student':std})

@login\_required

def dashboard(request):

return render(request, 'StudentDashboard.html')

@login\_required

def detail(request, post\_id):

if (request.user.student.isStudent == True):

std = Student.objects.get(email=request.user.student.email)

post = Internship.objects.get(id=post\_id)

status = InternshipAppliedDB.objects.filter(internship\_id=post\_id, student\_id=std.id)

print("status: ", len(status))

return render(request, 'StudentCompanyViewDetails.html',

{'post': post, 'student': std, 'status': len(status)})

else:

messages.info(request, 'Please sign in as a student')

return redirect(auth\_student)

def Jaccard(x, y):

"""returns the jaccard similarity between two lists """

intersection\_cardinality = len(set.intersection(\*[set(x), set(y)]))

union\_cardinality = len(set.union(\*[set(x), set(y)]))

return intersection\_cardinality / float(union\_cardinality)

def internshipApplied(request, post\_id):

std = Student.objects.get(email=request.user.student.email)

post = Internship.objects.get(id=post\_id)

student\_id = std.pk

internship\_id = post.pk

student\_name = std.name

student\_email = std.email

student\_mob = std.resume.mob

# Add Jacard Similarity Here

matching = 0

x = list(std.resume.skills) #student skills

y= post.skills #comp skills

print("the skills of the company internship are: ",y)

ans = Jaccard(x, y)

matching=ans\*100;

status = "pending"

new\_apply = InternshipAppliedDB(internship\_id=internship\_id, student\_id=student\_id, student\_name = student\_name, student\_email = student\_email, student\_mob = student\_mob, matching = matching, status=status)

new\_apply.save()

# return redirect('/internships')

return redirect('detail', post\_id=post\_id)

**Chapter 7-Output screen**

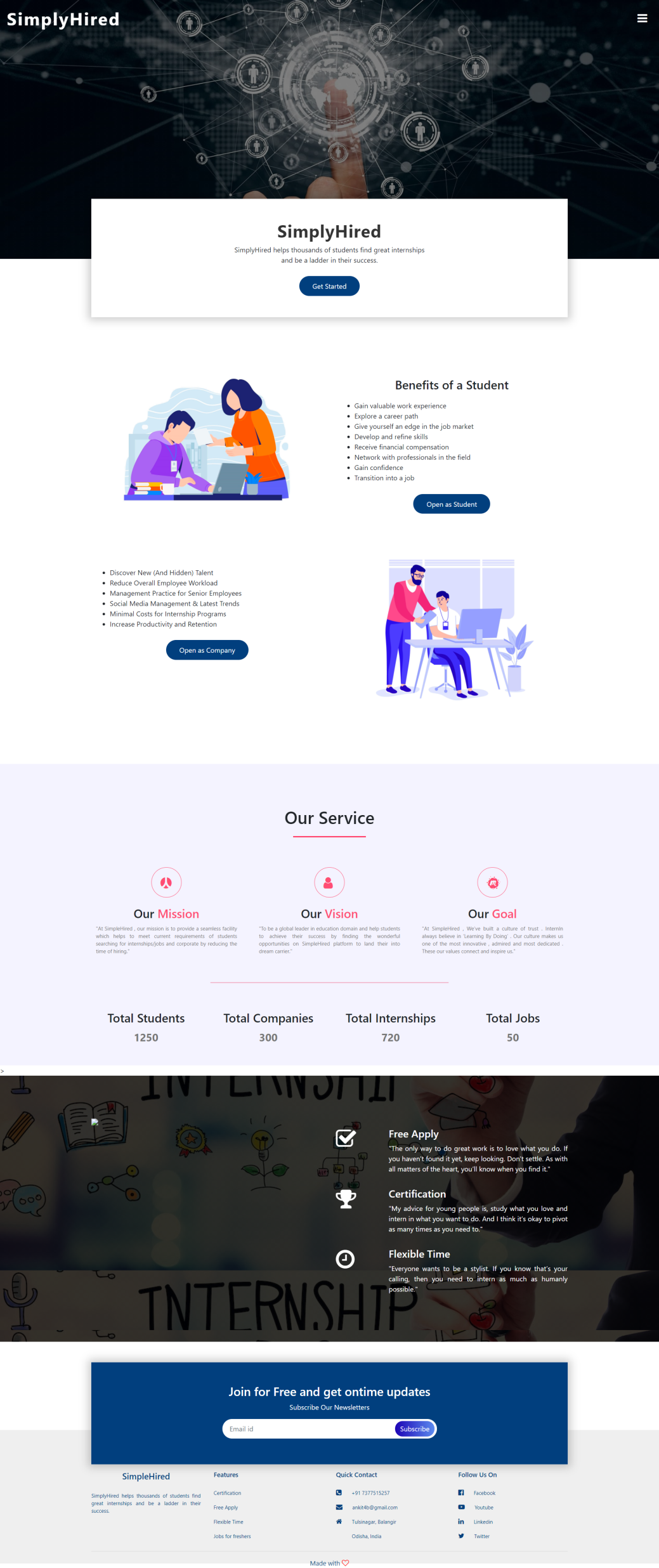


Figure 1

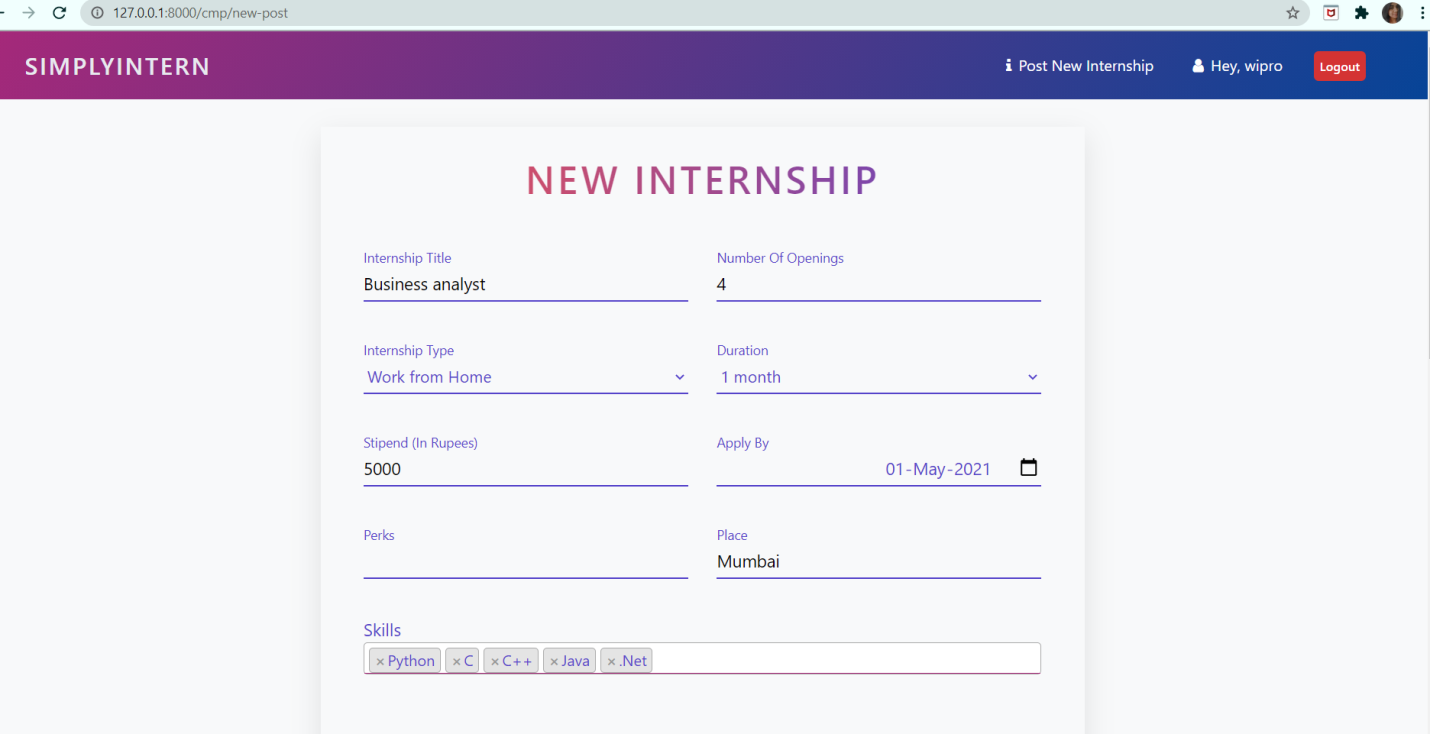


Figure 2

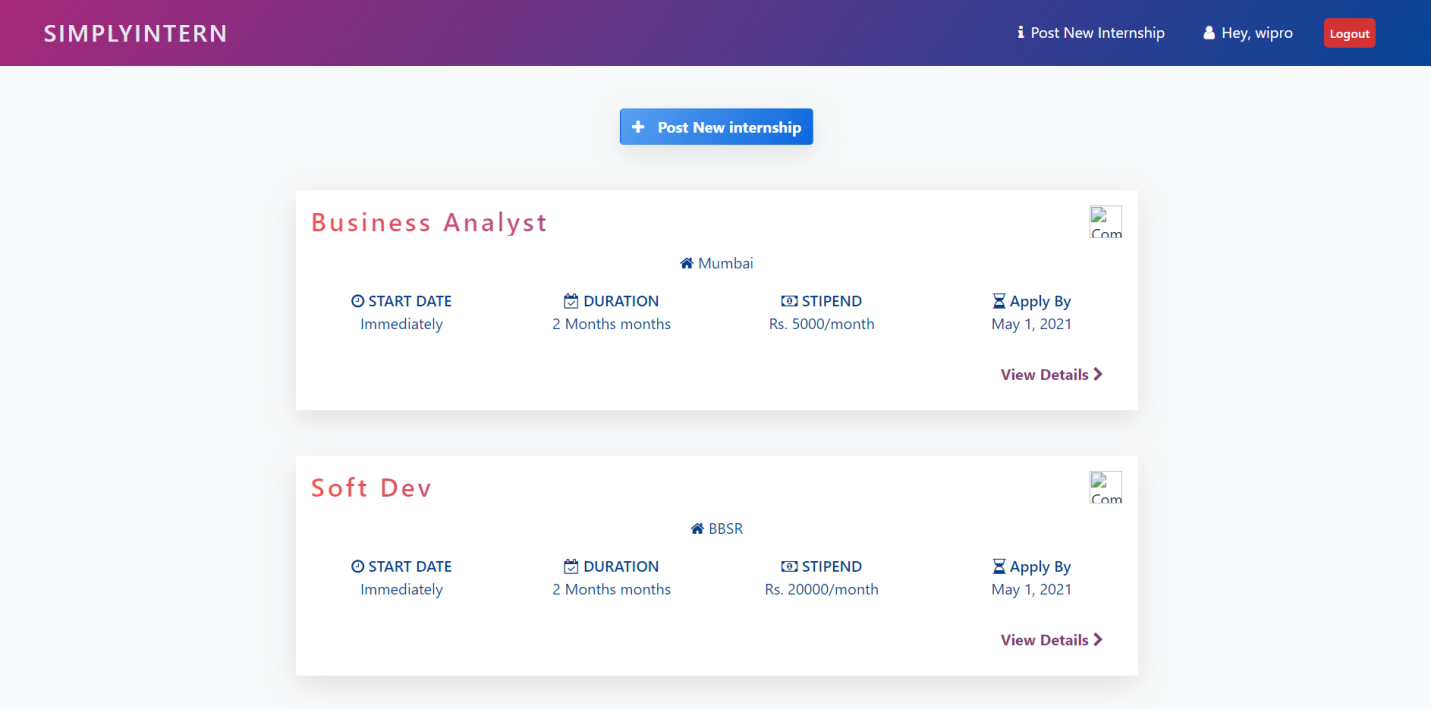


Figure 3

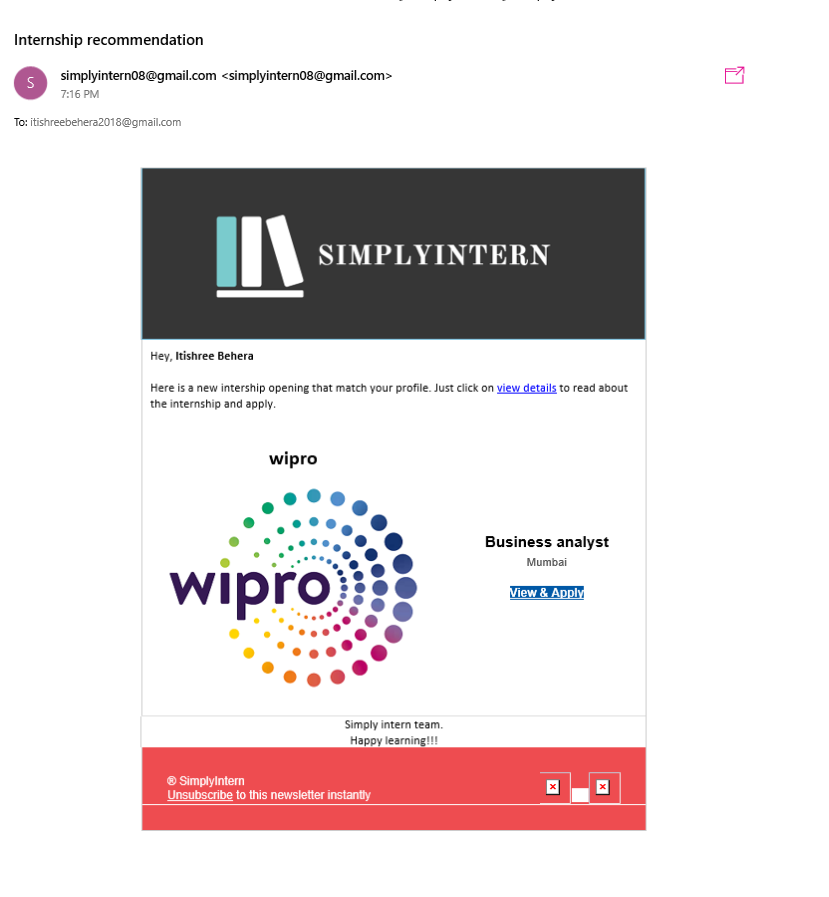


Figure 4

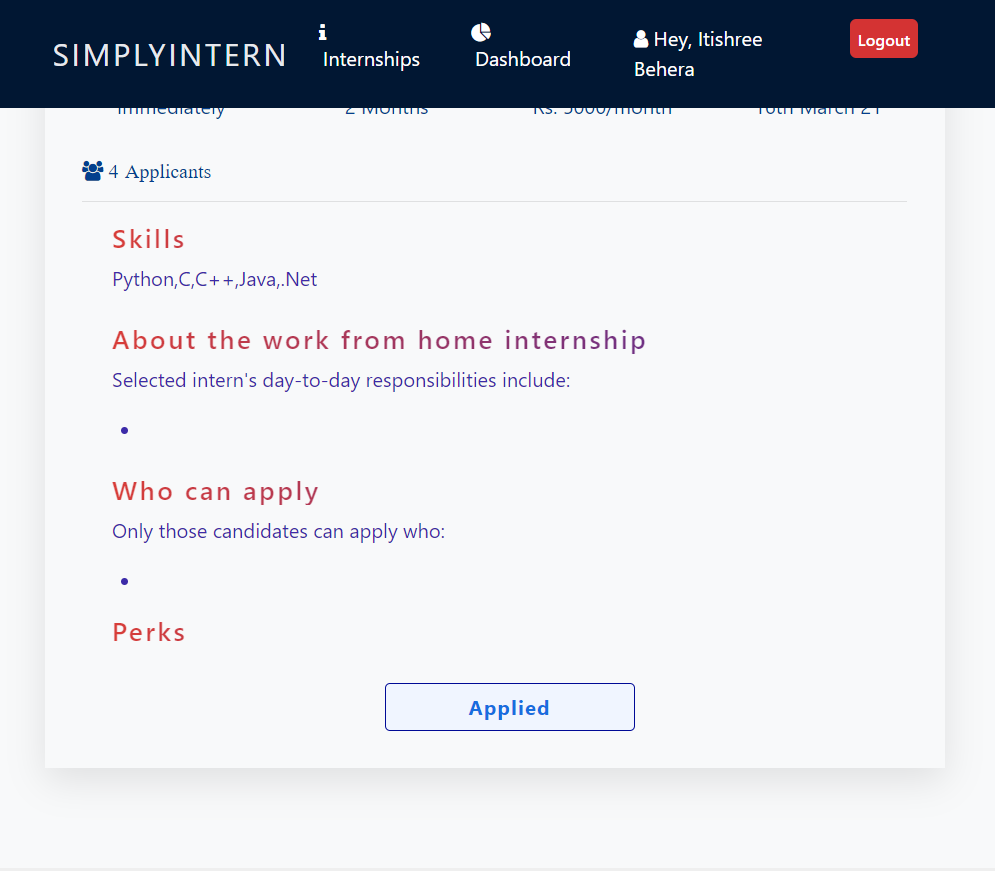


Figure 5

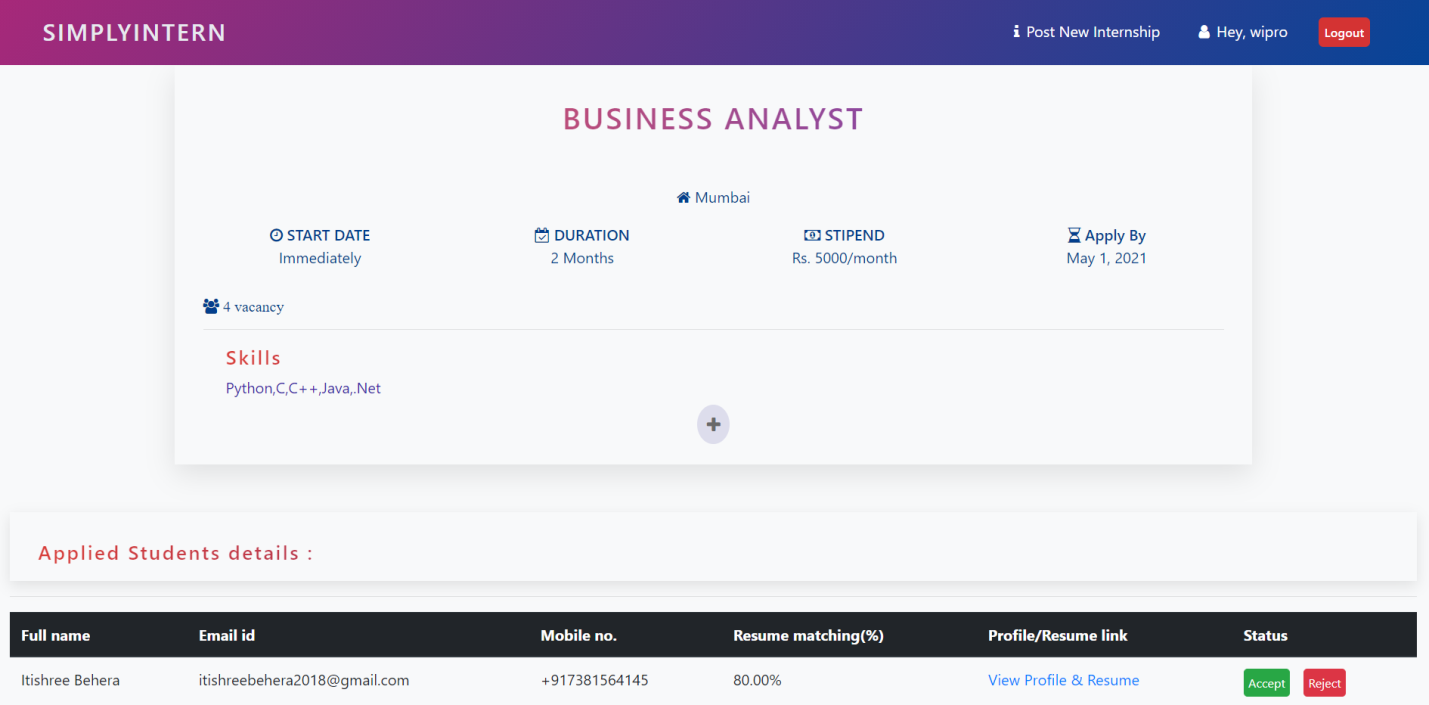


Figure 6

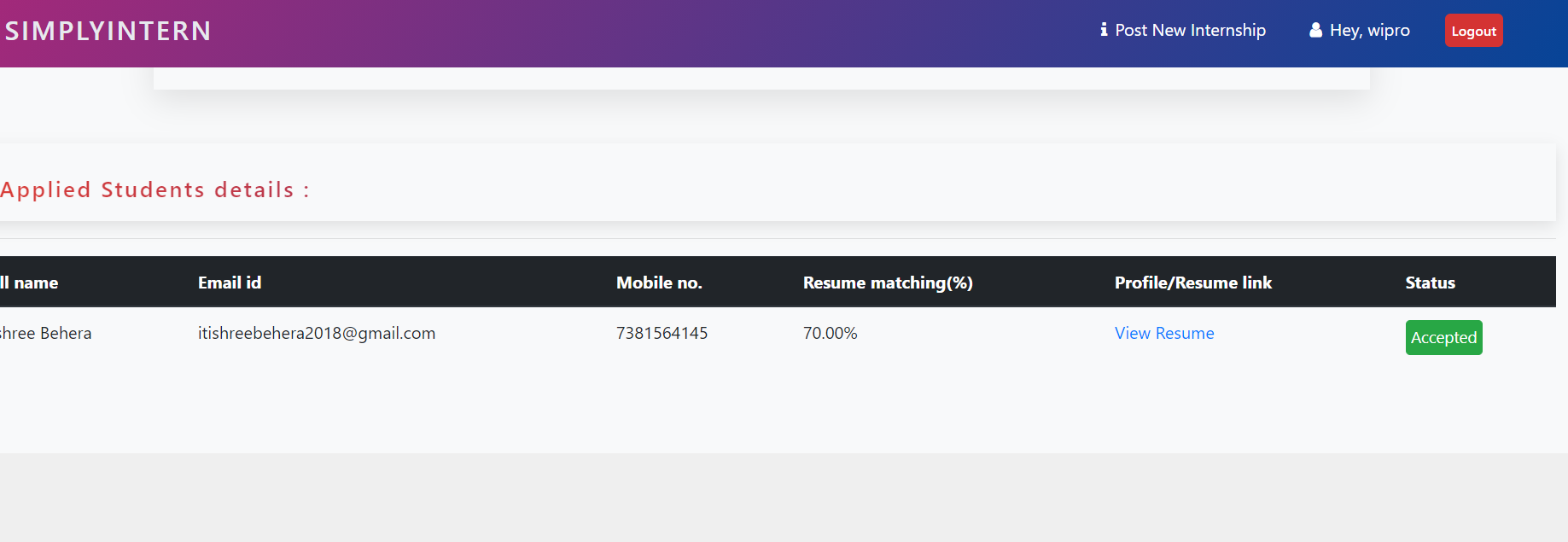


Figure 7



Figure 8

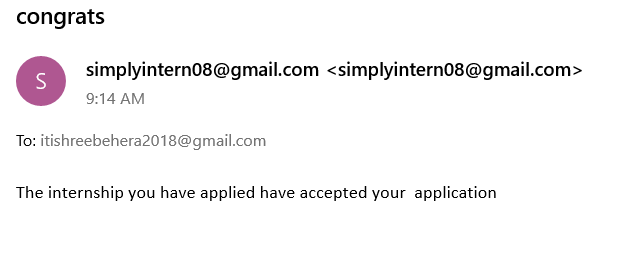


Figure 9

**Chapter 8. References/Bibliography**

1. Implementation of an Automated Job Recommendation System Based on Candidate Profiles Vinay Desai, Dheeraj Bahl, Shreekumar Vibhandik,Isra Fatma Prof. Vijay Kolekar , Dept. of Computer Engineering, K.J. College of Engg & Mgt. Research, Pune, India

2. Job Recommendation based on Job Seeker Skills: An Empirical Study Jorge Valverde-Rebaza Ricardo Puma Paul Bustios Nathalia C. Silva.

3. W. Hong, S. Zheng, H. Wang, J. Shi, “A Job Recommender System Based on User Clustering”, Journal of Computers, vol. 8, no. 8,pp. 1960- 1967, 1, Aug. 51 2013.

4. Diaby, M., E. Viennet, and T. Launay. Toward the next generation of recruitment tools: an online social networkbased job recommender system. in Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. 2013. ACM.

5. Yingya Zhang, Cheng Yang and Zhixiang NiuA, “Research of Job Recommendation System Based on Collaborative Filtering”, in Seventh International Symposium on Computational Intelligence and Design, 2014 IEEE DOI 10.1109/ISCID.2014.228

6. <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.jaccard_score.html>

7. <https://docs.djangoproject.com/>