“THE FREEBIRD”- A WEB-BASED APPLICATION WITH A JOB RECOMMENDATION SYSTEM

A CAPSTONE PROJECT REPORT

Submitted in partial fulfillment of the
requirement for the award of the
Degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE ENGINEERING WITH SPL. in DATA ANALYTICS

by

RASMIKA BILLA (18BCD7021)

Under the Guidance of

Dr. SANDIPAN MAITI

SCHOOL OF COMPUTERS ENGINEERING
VIT-AP UNIVERSITY
AMARAVATI- 522237

JANUARY 2021

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

A freelance worker or Freelancer is a person who is self-employed and not necessarily committed to a particular employer long-term. Freelancers are sometimes represented by a company or a temporary agency that resells freelance labor to clients, others work independently by using professional associations or websites to get work[1].

The workers are usually forced into nonstandard work arrangements, for others, loosening attachment to organizations provides the opportunities to accommodate their wants, preferences, and individualized lifestyles[2]. Thus, the application will predict the skill levels for various workers/customers based on their skill set and project work. This prediction will be in a generalized form of getting into freelancing and establishing a network. The app would allow users to post, manage and delete their services or requests.

In another scenario, if the customer wishes to search for jobs and wants to work within a community then the skillset of the customer can further be implemented in the developed job recommendation system to draw suitable job postings. The project work is for dual purposes helping the workers in either way out.

The problems discussed above have led to a major concern especially during times of pandemic and an immediate solution for this problem has become necessary in order to sustain the economic development.

1.1 OBJECTIVES

The following are the objectives of this project:

- Creating an application to provide effective assistance to people in search of work.
- Client and freelancer registration and login to make the application secure.
- Applying crud operations such as a client or a freelancer can add, post or delete services or requests.

- On the other end, any user can view all the services or any service details he/she is interested in.
- Predicting the best three job recommendations and required postings, if the user wishes to work for a company based on their qualification, location and work experience[3].
- Predicting the percentage chance of getting selected by depicting the scale of vacancies, recommendation index and application index.

1.2 BACKGROUND AND LITERATURE SURVEY

India had (according to the data provided by Mahesh Vyas of the Centre for Monitoring Indian Economy) around 403.5 million employed people and around 35 million (or 3.5 crores) openly unemployed people in the country just before the Covid-19 crisis, at the end of the 2019-20 financial year.

The unemployment rate in India rose sharply and touched a six-month high. The fact that the unemployment rate has been rising since September 2020 despite faster recovery has raised concerns. India's unemployment rate rose sharply to 9.1 percent in December 2020 even as economic activity continues to increase which is one of the major concerns. The project aims to solve the major concerns that are listed and is motivated to decrease unemployment.

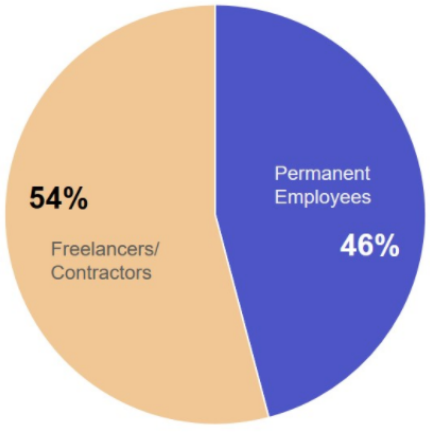

Figure 1. Google workforce of 220.000 workers

Google's workforce consisted of more than a lakh contractors/freelancers, which was more than the number of permanent employees on its payroll as of March 2019. The above figure depicts the workforce distribution at Google.

The rise in the adoption of remote teams has made freelancing a viable career path for skilled professionals in regions where domestic opportunities are low. The below figure amounts to the number of countries growing based on the freelancing markets. Among all these India ranks in the Top 10 countries with the fastest-growing earnings for freelancers.

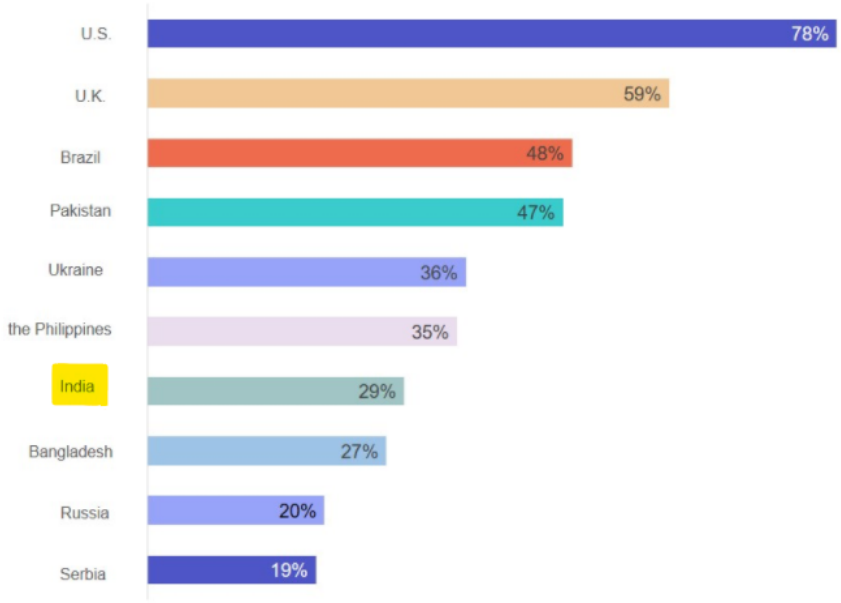

Figure 2. Fastest growing freelance markets

The data collected and analysed draws a decision that there is big concern among the markets in dual ways. The sectors are diverging into different kinds of markets where one is establishing productiving individually and the other within a community. To make balance between the both, the application developed provides a platform for the customers interested in freelancing as well as job postings[4]. A person who pursues individual productivity can go through freelancing and expertise in one to more fields whereas the other can look for best suitable job posts and apply for the ones where they have a higher chance of productivity.

1.3 ORGANIZATION OF THE REPORT

The remaining chapters of the project report are described as follows:

- Chapter 2 contains the proposed system, methodology, hardware, and software details.
- Chapter 3 gives a detailed explanation and implementation of the project.
- Chapter 4 discusses the results obtained after the project was implemented.
- Chapter 5 concludes the report.
- Chapter 6 consists of codes.
- Chapter 7 gives references.

CHAPTER 2

THE FREEBIRD APPLICATION AND JOB RECOMMENDATION SYSTEM

This Chapter describes the proposed system, working methodology, the dataset used, the pre-processing techniques applied on the dataset, software and hardware details of the project work.

2.1 PROPOSED APPLICATION/SYSTEM

The application is intended to help the users in trial ways. The user registered as a freelancer can use the web-based system to predict their skill level, post their services, manage their services, ad lastly have person requests as well. The user registered as client can easily view the services posted, and additionally can generate a customised requests explaining his/her requirements. The user wanting to work within a community can use the job recommendation system[5] to get their best three skills acknowledged and apply for the job posts recommended with a given percentage change of getting placed in that particular job.

2.2 WORKING METHODOLOGY

The application is connected to SQL[6] server which helps secure and maintain the data. All the webpages connected remain active till the work is executed. The prediction models are implemented behind the loop with reference to Flask for its integration.

2.2.1 DATASETS

The datasets for skill level prediction and job recommendation system are different and connected to the aspect of employment. The dataset analysed for predicting skill level of the user is FIVER dataset. The dataset has about six columns and eight thousand rows. The below figure.3 depicts a snapshot of the dataset. The second dataset implemented for the project work is JOB_POSTS dataset. The dataset consists of about twenty-four columns and thirty thousand rows. The below figure.4 depicts a snapshot of the dataset. Both the datasets are taken from Kaggle resource.

	Category	Subcat	name	price	stars	votes
0	Programming & Tech	Data Analysis & Reports	build automated and insightful power bi report	137.76	5.0	10
1	Lifestyle	Greeting Cards & Videos	get kermit to personalize a video birthday gre...	4.44	5.0	1k+
2	Programming & Tech	Website Builders & CMS	build or redesign your existing squarespace we...	66.66	5.0	68
3	Lifestyle	Cooking Lessons	create original eastern food cooking video recipe	17.78	4.9	5
4	Writing & Translation	Legal Writing	write gdpr privacy policy and terms and condit...	8.89	5.0	2

Figure 3. FIVER Dataset

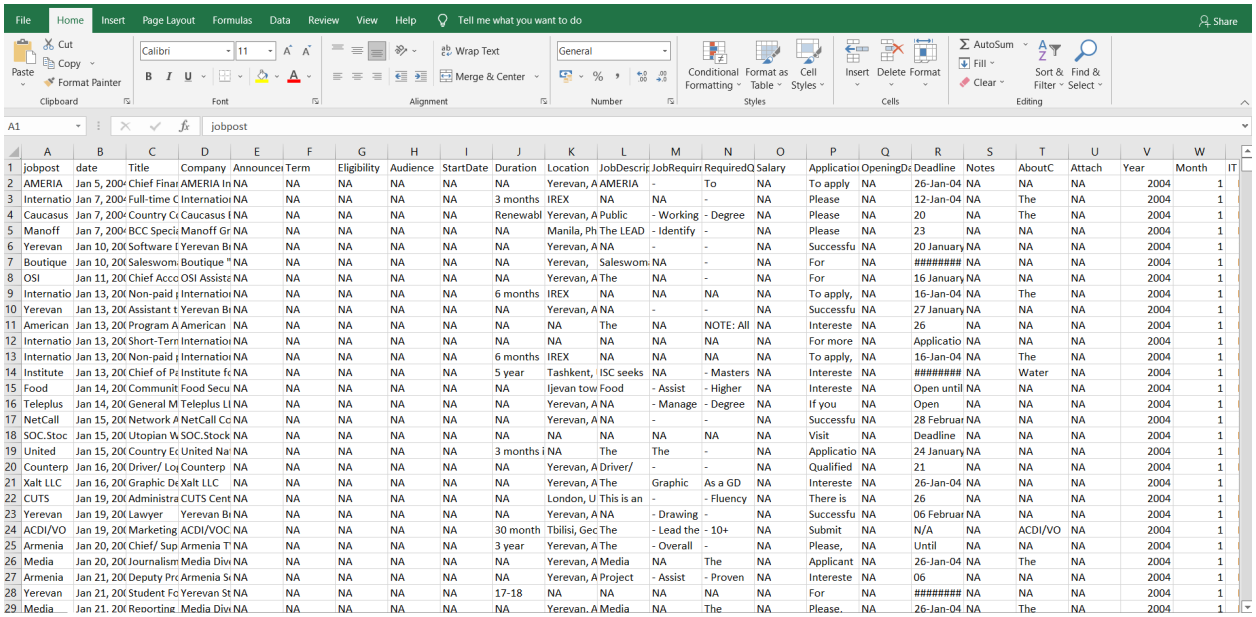

Figure 4. JOB_POSTS Dataset

2.2.2 PRE-PROCESSING TECHNIQUES

The technique used to modify the FIVER dataset is ordinal encoding[7] technique. The technique converts the unique categorical values to numerical values for better analysis. The figure below shows the implementation and output of ordinal encoding on the dataset.

```
from sklearn.preprocessing import OrdinalEncoder

ord_enc = OrdinalEncoder()

df['Category'] = ord_enc.fit_transform(df[['Category']])
```

```
df['Subcat'] = ord_enc.fit_transform(df[['Subcat']])
```

```
df
```

	Category	Subcat	stars
0	4.0	33.0	5.0
1	2.0	52.0	5.0
2	4.0	122.0	5.0
3	2.0	27.0	4.9
4	6.0	60.0	5.0

Figure 5. Ordinal Encoding

The JOB_POSTS dataset is the second data in which the technique used to preprocess is Customised Tokenizer[8] and Label Encoding. The technique is used to convert the textual data into simple format and extract the required skills and experience for predicting job roles for the user. The below figure is a code snippet implemented for converting textual data.

```
from nltk import word_tokenize
from nltk.stem import WordNetLemmatizer
class LemmaTokenizer(object):
    def __init__(self):
        # lemmatize text - convert to base form
        self.wnl = WordNetLemmatizer()
        # creating stopwords list, to ignore lemmatizing stopwords
        self.stopwords = stopwords.words('english')
    def __call__(self, doc):
        return [self.wnl.lemmatize(t) for t in word_tokenize(doc) if t not in self.stopwords]

# removing new line characters, and certain hypen patterns
df['RequiredQual']=df['RequiredQual'].apply(lambda x: x.replace('\n', ' ').replace('\r', '').replace('- ', ''). replace(' - ', ' to '))
```

Figure 6. Custom Tokenizer

2.3 SYSTEM DETAILS

This section describes the software and hardware details of the system:

2.3.1 SOFTWARE

- The frontend development for the application is completed with the help of HTML, CSS, and JavaScript as the programming languages.

- The backend development of the application is implemented using Php, MYSQL, and Flask for python integration.
- Google Colab is used for predicting and implementing various machine learning algorithms and Jupiter Notebook for developing the job recommendation system.
- Windows is used as the operating system.

2.3.2 HARDWARE

The programs were executed on a HP laptop with Intel core i5 8th generation CPU and 8 GB RAM with 256 GB SSD hard disk.

CHAPTER 3

DISCUSSION

The designs and pages are accomplished with HTML5[9] and CSS3[10]. The Index Page from which users can Login, Signup, Explore services, and go through contact and about web pages as well.

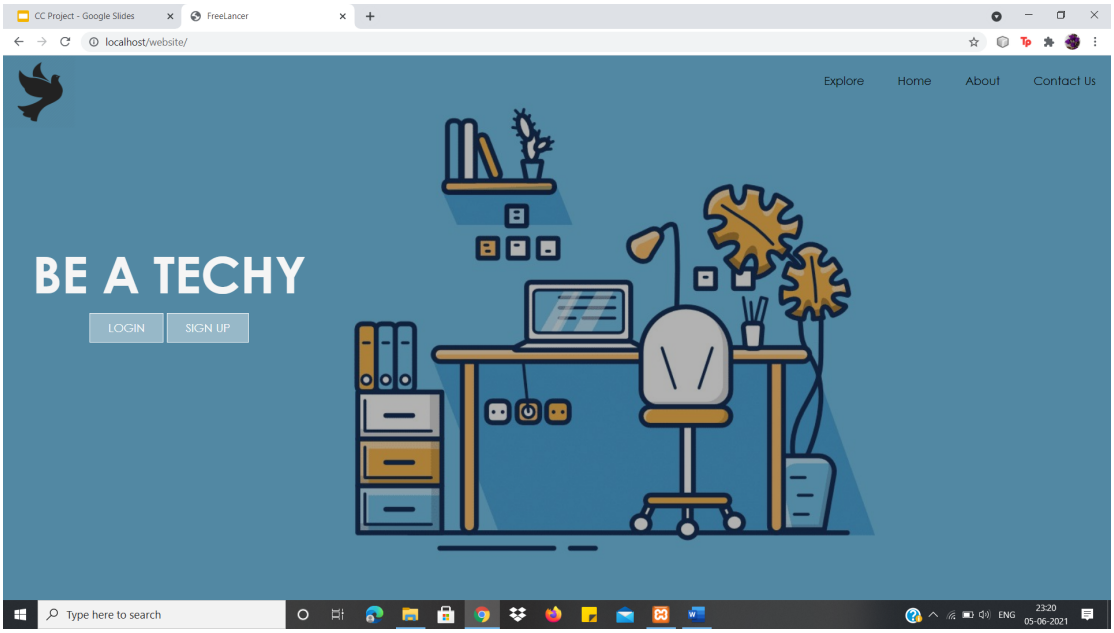

Figure 7. Index page

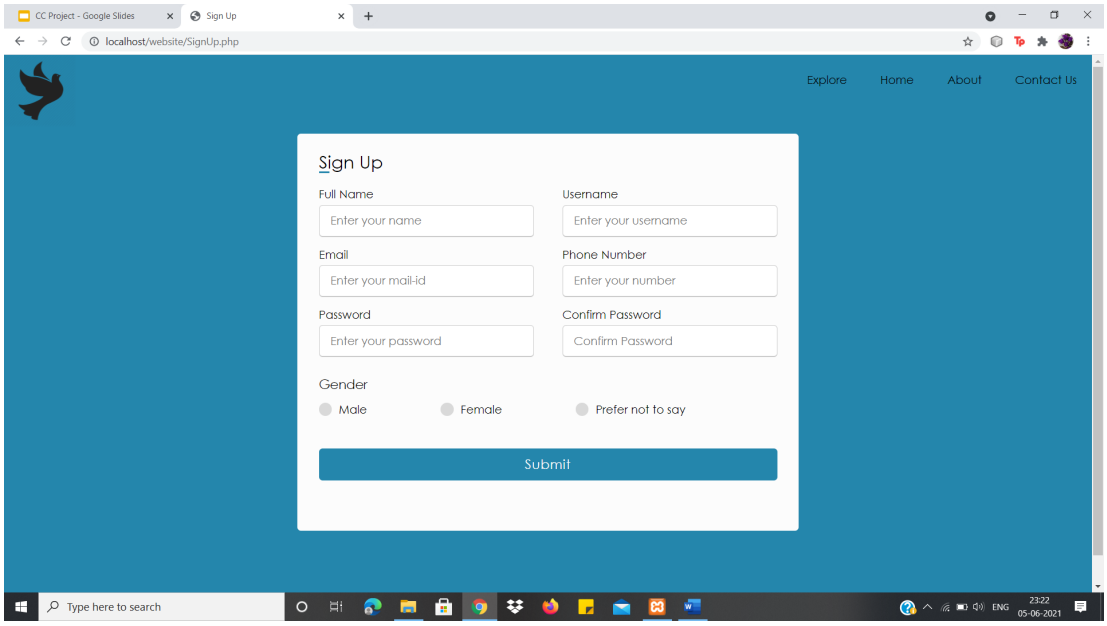

Figure 8. Signup Page

The user has to re-enter if the password and the confirm password is not matched. Also, if the username or password is incorrect the user is notified during the login process.

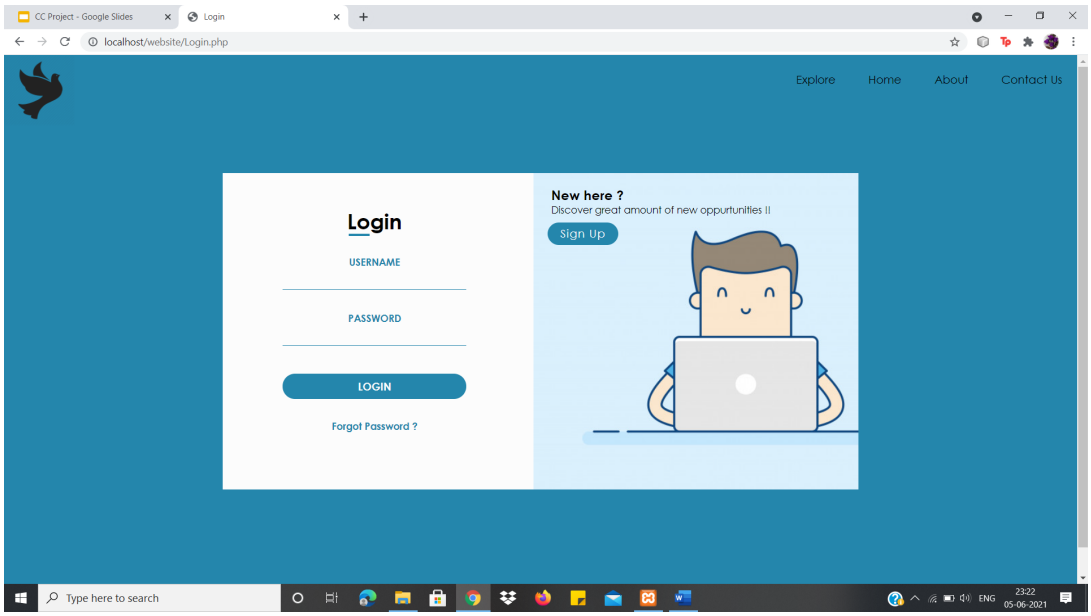

Figure 9. Login Page

The Home page where the user can either become a buyer or a seller depending on their necessities. The user can register for both as well.

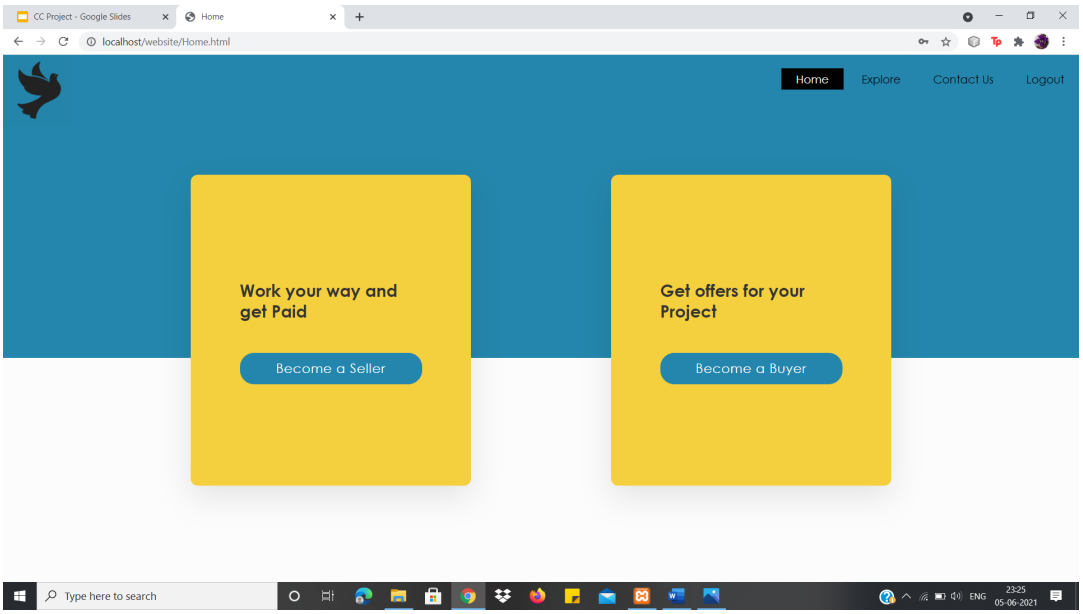

Figure 10. Home Page

The user registered as a Seller can go through their skill level, Publish/Post a service, manage their posts as well as manage the client requests.

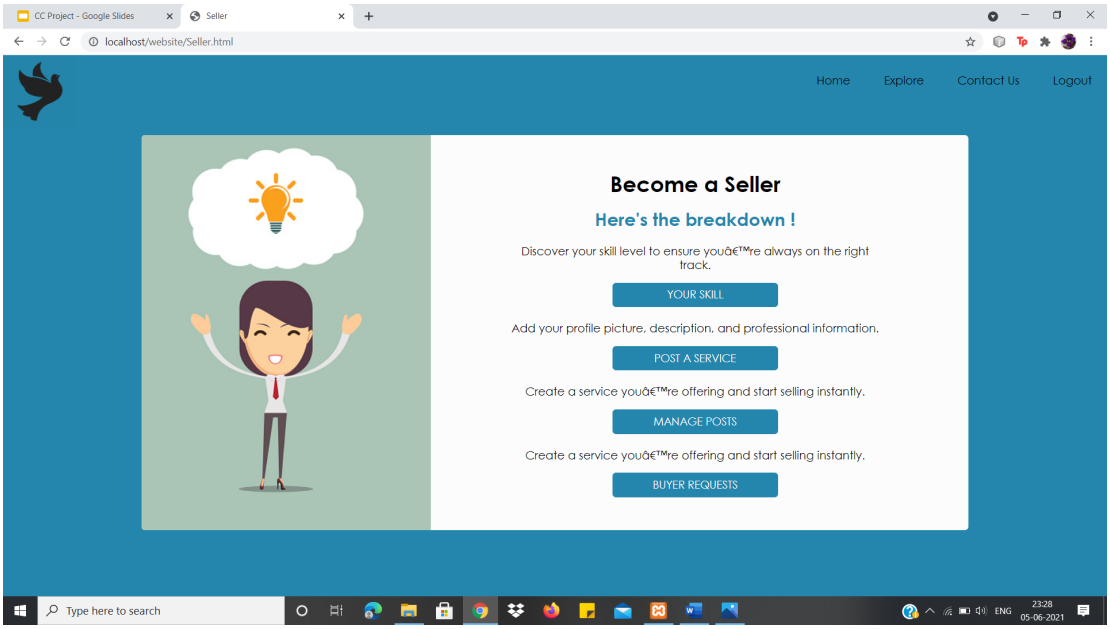

Figure 11. Freelancer main Page

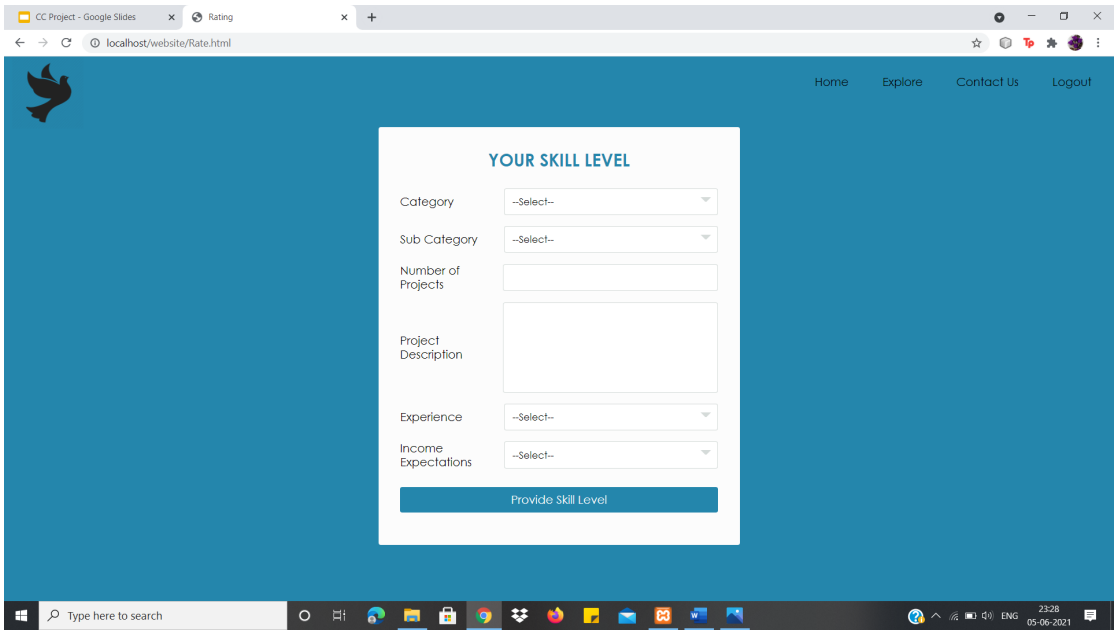

Figure 12. Predicting Skill Level

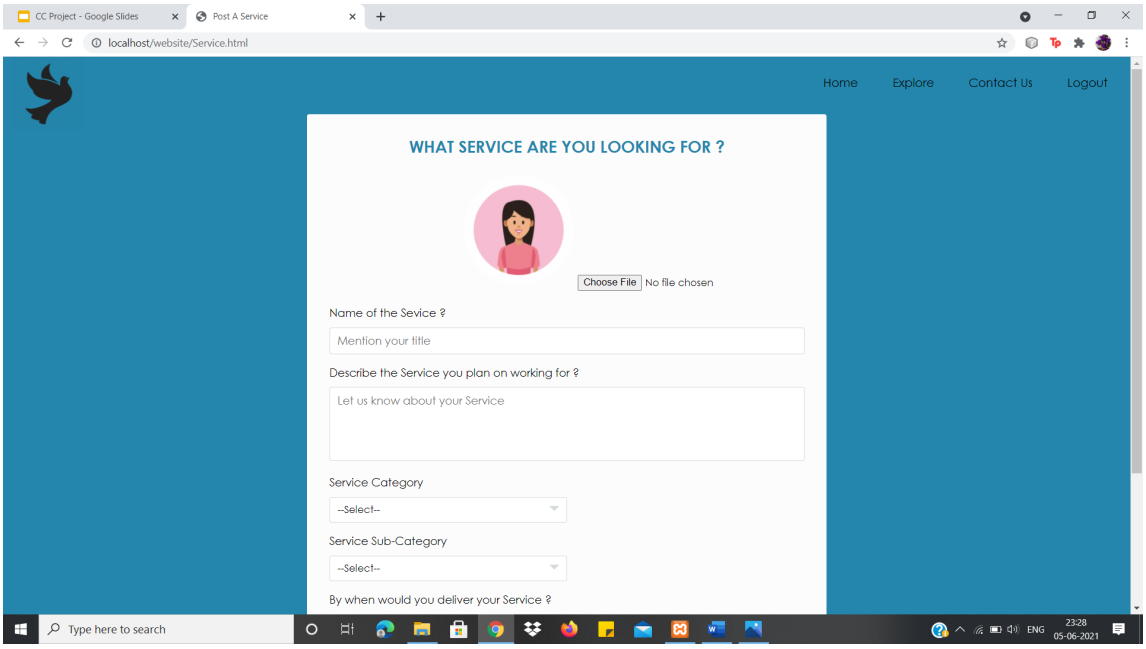

Figure 13. Posting services

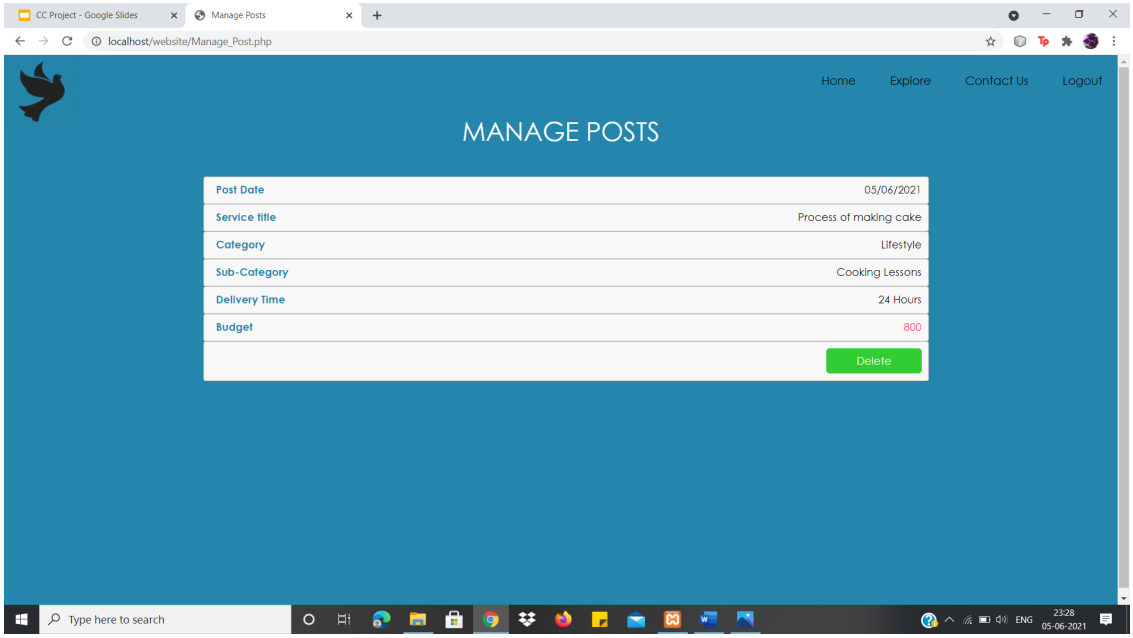

Figure 14. Managing the services posted

The user registered as a Buyer can publish/post a service request, manage their request posts as well as search and explore various services.

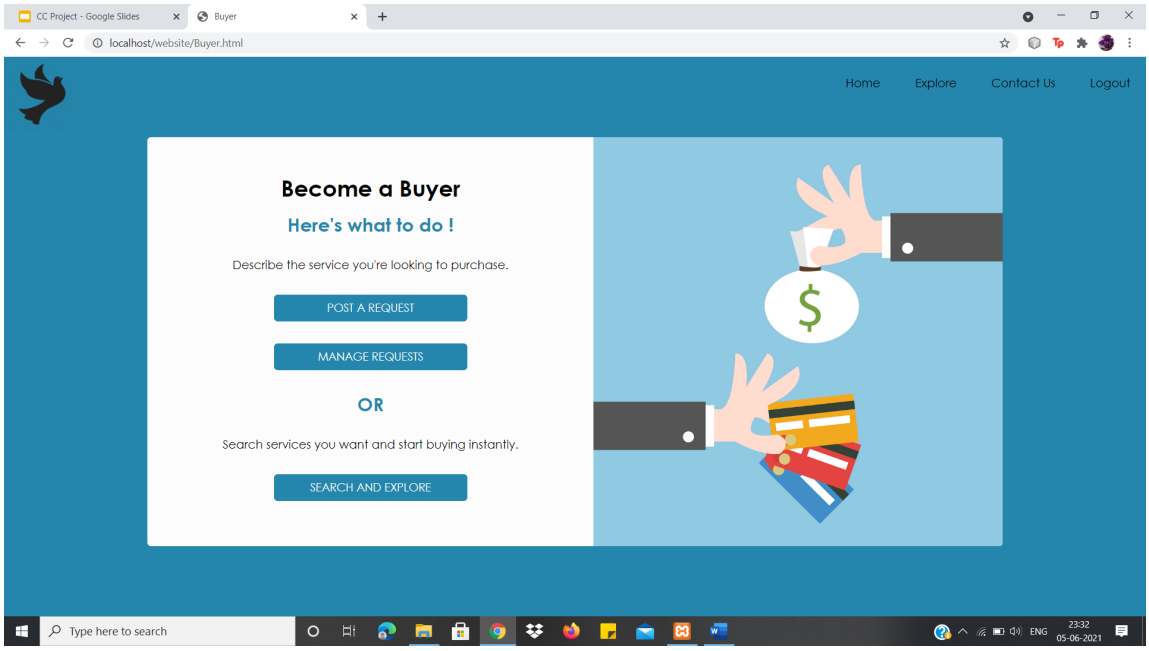

Figure 15. Clients main Page

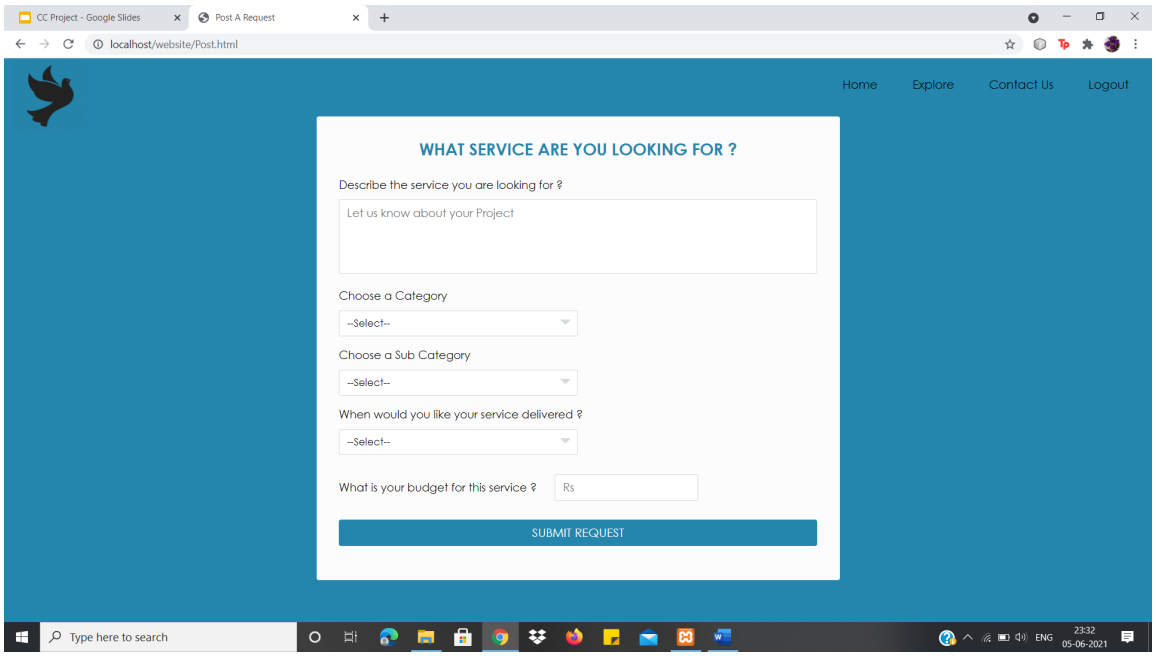

Figure 16. Client requests

The clients and freelancers can easily communicate in this way through the application developed and gain from the resources available. The data of the users is maintained in a secured manner as shown in the below figure.

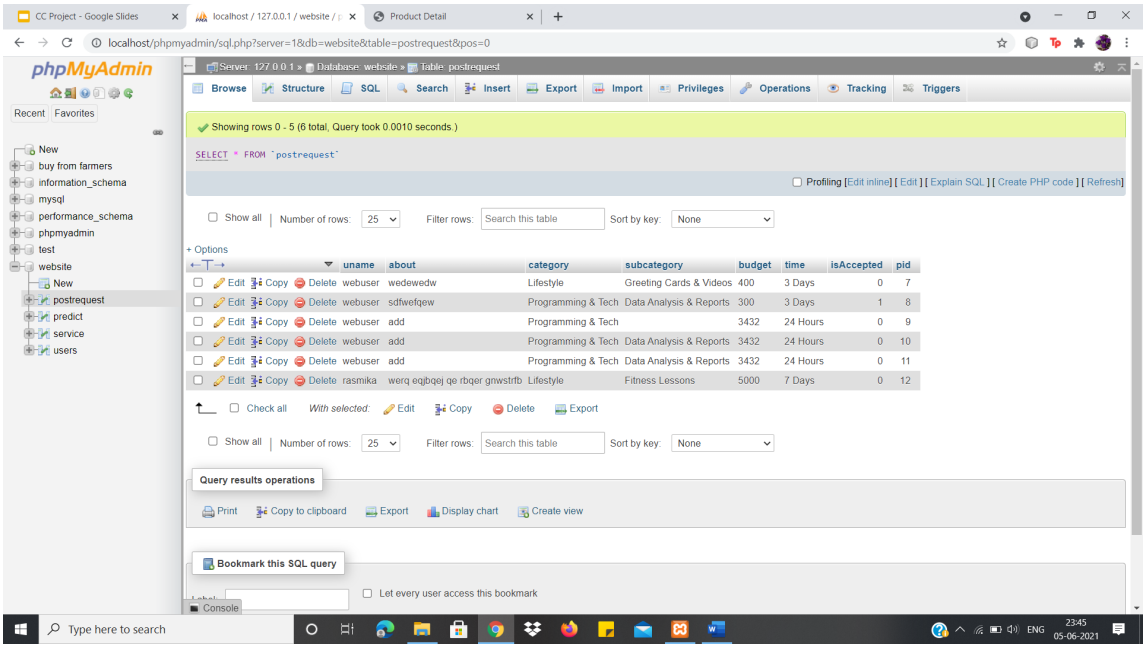

Figure 17. MySQL Server[11]

The user intending to join a community to earn a living can use the job recommendation system to get to his/her best three suitable job roles and their respective job postings that is developed by analysing the textual data with the help of natural language processing[12]. The analysis is based on the three narrowed prospects i.e required job qualification, skillset, experience, location, and vacancies. The below figure shows the recommended job titles for their respective work and experience with the specified job posting as per localities.

	Skill and Experience	1st Recommendation	2nd Recommendation	3rd Recommendation
0	Good knowledge of OOP and OOD; Experience in p...	Java Developer	Software Developer	Web Developer
1	Two years of experience in Java web programmin...	Java Developer	Software Developer	Web Developer
2	BS in Computer Science or any related technica...	.NET Developer	Web Developer	Software Developer
3	Excellent knowledge of PHP, MySQL's SQL, JavaS...	Web Developer	Java Developer	Software Developer
4	Minimum 2 years experience in relevant field; ...	Web Developer	Software Developer	Software Engineer
...	...	...	...	...
74	Positive attitude with a "can do" mentality; A...	iOS Developer	Web Developer	.NET Developer
75	Bachelor of Science degree in Computer Science...	Java Developer	Software Engineer	Software Developer
76	Bachelor's/Master's degree in CS or related di...	Software Developer	Software Engineer	Web Developer
77	Proficient in ASP.NET, C#, Java Script; And/or...	Programmer	Software Developer	Web Developer
78	Degree in Computer Science, Information Techno...	Java Developer	Software Developer	Software Engineer

79 rows × 4 columns

Figure 18. Job role recommendations

	jobpost	date	Title	Company
2158	Essence Development LLC\nTITLE: Tester/ Quali...	Feb 16, 2006	Tester/ Quality Assurance Engineer	Essence Development LLC
3138	Essence Development LLC\nTITLE: Tester/ Quali...	Jan 8, 2007	Tester/ Quality Assurance Engineer	Essence Development LLC
3295	Essence Development LLC\nTITLE: Tester/ Quali...	Feb 12, 2007	Tester/ Quality Assurance Engineer	Essence Development LLC
3518	Essence Development LLC\nTITLE: Tester/ Quali...	Apr 2, 2007	Tester/ Quality Assurance Engineer	Essence Development LLC

Figure 19. Job posts based on role, locality and vacancies[13]

3.1 ML ANALYSIS

The accuracy scores for predicting the skill level is shown in the below table.

Model	Accuracy Score
1. K-nearest neighbours	76.21%
2. Support Vector Machine[14]	72.89%
3. Decision Tree	92.94%
4. Random Forest	96.78%
5. Gradient Boosting	84.78%
6. XGB Boost	81.29%
7. Naive Bayes Gaussian Classifier[15]	86.34%
8. Naive Bayes Multinomial Classifier	76.21%

Table 1. Models implemented and their accuracies(1)

The accuracy scores for implementing the job recommendation system is shown in the below table.

Model	Accuracy Score
1. K-nearest neighbours[16]	67.29%
2. Support Vector Machine	77.21%
3. Decision Tree	78.83%
4. Random Forest	81.09%
5. Gradient Boosting	46.45%
6. XGB Boost	56.89%
7. Naive Bayes Gaussian Classifier	62.54%
8. Naive Bayes Multinomial Classifier	69.92%
9. Logistic Regression	95.67%
10. AdaBoost + Random Forest	83.54%
11. Single Classifier	91.29%
12. Meta Classifier	80.20%

Table 2. Models implemented and their accuracies (2)

The highest accuracy score for skill level prediction is given by Random Forest Algorithm[17], whereas for the job recommendation system, Logistic Regression[18] shows an eminent progress. Even though the complex algorithms such as bagging and boosting[19] are applied there is no improvement in the accuracy. The reason behind the situation is that the data provided is textual and requires NLP techniques with Logistic regression[20]. Also, while predicting skill level the data is converted from categorical to numerical making the Random Forest algorithm more dominant.

CHAPTER 4

RESULTS

The application consists of multiple web pages allowing the user/freelancer to go through skill prediction, post some specific ones with valid details and manage their posts as well as accept or deny client requests. Also, when an invalid detail is/are submitted, an error message is displayed while registration and signing in to the website. At the same time, the clients can explore all the services, post a specific request if they need with intended skill, time and payment.

The website is successfully hosted and remains active till the work is executed. The job recommendation system helps the users to find the jobs they intend to work for with the simplistic algorithms and implementations. All the predicting models and recommendation system are integrated with the web application.

The project abides by the objectives discussed above and completes the entire process with proper work flow in the designated time.

CHAPTER 5

CONCLUSION & FUTURE WORKS

The application builds a path to utilize every second to its maximum. It allows the clients and the freelancers to make the necessary arrangements for solving the issues discussed above as far as possible to ensure proper economic growth and stability in India. These small yet deciding issues with rising technology, both are open to improvements with huge scopes. Lot can be done in this area.

This project can be extended as follows:

- to support people to earn a living
- to make the website available worldwide
- to ensure proper communication between clients and the freelancers
- to make the portal much more efficient and secure by using advance technologies[21]
- to make recommendation more efficient with vacancy data list[22]

CHAPTER 6

APPENDIX

Python Code: (Skill Level Prediction)

```
import pandas as pd
import numpy as np
import pickle

df=pd.read_csv(r'C:/Users/RASMIKA BILLA/Downloads/fiverr_clean.csv')
df.drop('name',axis=1,inplace=True)
df.drop('price',axis=1,inplace=True)
df.drop('votes',axis=1,inplace=True)
df= df.dropna()

from sklearn.preprocessing import OrdinalEncoder
ord_enc = OrdinalEncoder()
df['Category'] = ord_enc.fit_transform(df[['Category']])
df['Subcat'] = ord_enc.fit_transform(df[['Subcat']])
df['Project'] = ord_enc.fit_transform(df[['Project']])
df['Experience'] = ord_enc.fit_transform(df[['Experience']])

X= df.drop('stars',axis=1)
y=df['stars']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test= train_test_split(X, y, test_size=0.33)

from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators = 10, random_state = 0)
regressor.fit(X,y)

pickle.dump(regressor, open('data.pkl','wb'))
```

Python Code: (Using Flask and integrating with the web page)

```
from flask import Flask, render_template, request
import pickle
import numpy as np
```

```
app = Flask(__name__)

model=pickle.load(open('data.pkl','rb'))

@app.route('/')
def hello_world():
        return render_template('Rate.html')

@app.route('/predict', methods=['POST'])
def predict():
        data1=request.form.get('slct1')
        data2=request.form.get('slct2')
        data3=request.form.get('proj')
        data4=request.form.get('income')
        final=np.array([[data1,data2]])
        prediction=model.predict(final)
        print(prediction)
        if prediction>=str(4.5):
                return render_template('Rate.html',predict='Good')
        else if prediction>=str(3.5) & prediction<str(4.5):
                return render_template('Rate.html',predict='Average')
        else:
                return render_template('Rate.html',predict='Needs improvement')

if __name__=="__main__":
        app.run(debug=True)
```

Python Code: (Job Recommendation System)

```
import numpy as np
import pandas as pd

data = pd.read_csv('job posts.csv')
data.head()

for c in data.columns:

  print(data[c].value_counts().to_frame())
```

```
data.isnull().any()

from sklearn.feature_extraction import DictVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split

import nltk
from nltk.corpus import stopwords
from sklearn.preprocessing import LabelEncoder
from sklearn.mixture import GaussianMixture
from sklearn.cluster import KMeans
from sklearn.metrics import accuracy_score, auc, roc_curve, roc_auc_score

#IT Jobs
df = data[data['IT']]
#selecting
cols = ['RequiredQual', 'Eligibility', 'Title', 'JobDescription', 'JobRequirment']
df=df[cols]
df.head(5)

classes = df['Title'].value_counts()[:21]
keys = classes.keys().to_list()

df = df[df['Title'].isin(keys)]
df['Title'].value_counts()

def chane_titles(x):
    x = x.strip()
    if x == 'Senior Java Developer':
        return 'Java Developer'
    elif x == 'Senior Software Engineer':
        return 'Software Engineer'
    elif x == 'Senior QA Engineer':
        return 'Software QA Engineer'
    elif x == 'Senior Software Developer':
        return 'Senior Web Developer'
    elif x =='Senior PHP Developer':
        return 'PHP Developer'
    elif x == 'Senior .NET Developer':
        return '.NET Developer'
```

```
    elif x == 'Senior Web Developer':
        return 'Web Developer'
    elif x == 'Database Administrator':
        return 'Database Admin/Dev'
    elif x == 'Database Developer':
        return 'Database Admin/Dev'

    else:
        return x


df['Title'] = df['Title'].apply(chane_titles)
df['Title'].value_counts()

from nltk import word_tokenize
from nltk.stem import WordNetLemmatizer
class LemmaTokenizer(object):
    def __init__(self):
        # lemmatize text - convert to base form
        self.wnl = WordNetLemmatizer()
        # creating stopwords list, to ignore lemmatizing stopwords

        self.stopwords = stopwords.words('english')
    def __call__(self, doc):
        return [self.wnl.lemmatize(t) for t in word_tokenize(doc) if t not in self.stopwords]

# removing new line characters, and certain hypen patterns

df['RequiredQual']=df['RequiredQual'].apply(lambda x: x.replace('\n', ' ').replace('\r', '').replace('-
', ''). replace(' - ', ' to '))

import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('corpus')
nltk.download('wordnet')

from sklearn.feature_extraction.text import TfidfVectorizer

y = df['Title']
```

```
X = df['RequiredQual']

vectorizer = TfidfVectorizer(tokenizer=LemmaTokenizer(), stop_words='english')
vectorizer.fit(X)

tfidf_matrix = vectorizer.transform(X)

X_tdif = tfidf_matrix.toarray()

enc = LabelEncoder()
enc.fit(y.values)
y_enc=enc.transform(y.values)

X_train_words, X_test_words, y_train, y_test = train_test_split(X, y_enc, test_size=0.15,
random_state=10)

X_train = vectorizer.transform(X_train_words)
X_train = X_train.toarray()

X_test = vectorizer.transform(X_test_words)
X_test = X_test.toarray()

from sklearn.naive_bayes import GaussianNB, MultinomialNB
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
gnb = GaussianNB()
train_preds = gnb.fit(X_train, y_train).predict(X_train)
test_preds = gnb.predict(X_test)

print('Train acc: {0}'.format(accuracy_score(y_train, train_preds)))
print('Test acc: {0}'.format(accuracy_score(y_test, test_preds)))

from sklearn import svm
clf_svm = svm.SVC(kernel='linear')
train_preds = clf_svm.fit(X_train, y_train).predict(X_train)
test_preds = clf_svm.predict(X_test)

print('Train acc: {0}'.format(accuracy_score(y_train, train_preds)))
print('Test acc: {0}'.format(accuracy_score(y_test, test_preds)))

from sklearn.tree import DecisionTreeClassifier
```

```
from sklearn import metrics
DT = DecisionTreeClassifier(random_state=0)
train_preds = DT.fit(X_train, y_train).predict(X_train)
test_preds = DT.predict(X_test)

print('Train acc: {0}'.format(accuracy_score(y_train, train_preds)))
print('Test acc: {0}'.format(accuracy_score(y_test, test_preds)))

from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
RF = RandomForestClassifier(random_state=0)
train_preds = RF.fit(X_train,y_train).predict(X_train)
test_preds = RF.predict(X_test)

print('Train acc: {0}'.format(accuracy_score(y_train, train_preds)))
print('Test acc: {0}'.format(accuracy_score(y_test, test_preds)))

from sklearn.linear_model import LogisticRegression

logistic = LogisticRegression(max_iter=15,verbose=1, C=0.75)

train_preds = logistic.fit(X_train, y_train).predict(X_train)
test_preds = logistic.predict(X_test)

print('Train acc: {0}'.format(accuracy_score(y_train, train_preds)))
print('Test acc: {0}'.format(accuracy_score(y_test, test_preds)))

from sklearn import svm
clf_svm = svm.SVC(kernel='linear')
clf_svm.fit(x_train, y_train)
y_pred = clf_svm.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

from sklearn.ensemble import GradientBoostingClassifier
from sklearn import metrics
clf = GradientBoostingClassifier()
clf = clf.fit(x_train,y_train)
y_pred = clf.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
import xgboost as xgb
from sklearn import metrics
clf = xgb.XGBClassifier()
clf = clf.fit(x_train,y_train)
y_pred = clf.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

from sklearn.ensemble import RandomForestClassifier
clf=RandomForestClassifier(n_estimators=100)
clf.fit(x_train,y_train)
y_pred=clf.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
y_pred = gnb.fit(x_train, y_train).predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

from sklearn.naive_bayes import MultinomialNB
clf = MultinomialNB()
clf.fit(x_train,y_train)
y_pred=clf.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

RF=RandomForestClassifier()
Abc=AdaBoostClassifier(base_estimator=RF)     #ada
bag_clf=BaggingClassifier(base_estimator=Abc)  #bagging
bag_clf.fit(x_train, y_train)
y_pred=bag_clf.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

from sklearn.ensemble import StackingClassifier
from mlxtend.classifier import StackingClassifier
meta=StackingClassifier(classifiers=[DT,RF],meta_classifier=LR)
meta.fit(x_train, y_train)
y_pred=meta.predict(x_test)
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
preds_data = {'Skill and Experience': [], '1st Recommendation': [], '2nd Recommendation': [],
'3rd Recommendation': []}
y_preds_proba = logistic.predict_proba(X_test)

counter = 0
for idx, (pred_row, true_job_position) in enumerate(zip(y_preds_proba, y_test)):
    class_preds = np.argsort(pred_row)

    # delete true class
    for i in [-1, -2]:
        if class_preds[i] == true_job_position:
            class_preds=np.delete(class_preds,i)

    # getting other 2 highest job predictions
    top_classes = class_preds[-2:]

    # obtaining class name string from int label
    class_names = enc.inverse_transform(top_classes)
    true_job_position_name = enc.inverse_transform([true_job_position])

    # saving to dict
    preds_data['Skill and Experience'].append(X_test_words.iloc[idx])
    preds_data['1st Recommendation'].append(true_job_position_name[0])
    preds_data['2nd Recommendation'].append(class_names[1])
    preds_data['3rd Recommendation'].append(class_names[0])

preds_df = pd.DataFrame.from_dict(preds_data)
#preds_df.to_csv('Recommendations.csv', index=False)
preds_df

data.loc[data['Title'] == "Tester/ Quality Assurance Engineer"]
%chance = (vacancies + (appl_idx/2) - recommendation_index)/vacancies * 100
%chance
```

Frontend Code (Home.html)

```
<!DOCTYPE html>
<html>
<head>
        <title>Home</title>
```

```
        <link rel="stylesheet" type="text/css" href="css/home.css">
</head>
<body>

        <header>
                <div class="main">
                        <div class="logo">
                                <img src="5.jpg">
                        </div>
                        <ul>
                                <li class="active"><a href="Home.html">Home</a></li>
                                <li><a href="Explore.php">Explore</a></li>
                                <li><a href="ContactUs.html">Contact Us</a></li>
                                <li><a href="index.html">Logout</a></li>
                        </ul>
                </div>
    </header>

<section>
        <div class="container">
                <div class="box">
                        <h2>01</h2>
                        <h3>Work your way and get Paid</h3>
                        <div class="button">
                                <a href="Seller.html"><input type="submit" value="Become a
Freelancer"></a>
                        </div>
                </div>
                <div class="box">
                        <h2>02</h2>
                        <h3>Get offers for your Project</h3>
                        <div class="button">
                                <a href="Buyer.html"><input type="submit" value="Become a
Client"></a>
                        </div>
                </div>
        </div>
</section>

</body>
```

</html>

Frontend Code (service.css)

```
*{
        margin:0;
        padding:0;
        box-sizing: border-box;
        font-family: Century Gothic;
}

body{
        padding:0 10px;
        background-color: #2486AC;
}

header{
        background-color: #2486AC;
        background-size: contain;
        height: 5vh;
        background-position: center;
}

ul{
        float:right;
        list-style-type: none;
        margin-top: 25px;
}

ul li{
        display:inline-block;
}

ul li a{
        text-decoration: none;
        color: #000;
        padding: 5px 20px;
        border: 1px solid transparent;
        transition: 0.6s ease;
}
```

```
ul li a:hover{
	background-color: #000;
	color:#fff;
}

ul li.active a{
	background-color: #000;
	color:#fff;
}

.logo img{
	float:left;
	width:100px;
	height:auto;
}

.wrapper{
	max-width: 700px;
	width: 100%;
	background: #fcfcfc;
	margin: 40px auto;
	padding: 30px;
	border-radius: 3px;
}

.wrapper .title{
	font-size: 22px;
	font-weight: 700;
	margin-bottom: 25px;
	color: #2486AC;
	text-align: center;
	text-transform: uppercase;
}

.wrapper .form{
	width: 100%;
}

.wrapper .form .service-pic{
	margin-bottom: 20px;
```

```
}

.wrapper .form .service-pic img{
        height: 150px;
        width: 150px;
        border-radius: 50%;
        margin-left: 180px;
}

.wrapper .form .input_field{
        margin-bottom: 15px;
}

.wrapper .form .input_field label{
        width: 100%;
        font-size: 15px;
}

.wrapper .form .input_field .textarea{
        width: 100%;
        outline: none;
        margin-top: 10px;
        border: 1px solid #d5dbd9;
        font-size:15px;
        padding: 8px 10px;
        border-radius: 3px;
        transition: all 0.3s ease;
}

.wrapper .form .input_field .input{
        width: 100%;
        outline: none;
        margin-top: 10px;
        border: 1px solid #d5dbd9;
        font-size:15px;
        padding: 8px 10px;
        border-radius: 3px;
        transition: all 0.3s ease;
}
```

```
.budget{
        width: 30%;
        outline: none;
        margin-top: 10px;
        margin-left: 20px;
        border: 1px solid #d5dbd9;
        font-size:15px;
        padding: 8px 10px;
        border-radius: 3px;
        transition: all 0.3s ease;
}

.wrapper .form .input_field .textarea{
        resize: none;
        height: 100px;
}

.wrapper .form .input_field .custom_select{
        position: relative;
        margin-top: 10px;
        width: 50%;
        height: 35px;
}

.wrapper .form .input_field .custom_select select{
        appearance: none;
        border: 1px solid #d5dbd9;
        width: 100%;
        height: 100%;
        padding: 8px 10px;
        border-radius: 3px;
        outline: none;
}

.wrapper .form .input_field .custom_select:before{
        content:"";
        position: absolute;
        top:12px;
        right: 10px;
        border: 7px solid;
```

```
        border-color: #d5dbd9 transparent transparent transparent;
        pointer-events: none;
}

.wrapper .form .input_field .input:focus,
.wrapper .form .input_field .textarea:focus,
.wrapper .form .input_field select:focus{
        border: 1px solid #2486AC;
}

.wrapper .form .input_field .btn{
        width: 100%;
        padding: 8px 10px;
        margin-top: 10px;
        font-size: 16px;
        border: 0;
        background:#2486AC;
        color: #fcfcfc;
        cursor: pointer;
        border-radius: 3px;
        outline: none;
        text-transform: uppercase;
}
```

Backend Code (Predict.php)

```
<?php
include 'connect_db.php';
session_start();
$uname = $_SESSION['user'];
$category = $_POST['slct1'];
$subcategory = $_POST['slct2'];
$projects = $_POST['projects'];
$project_description = $_POST['project_description'];
$experience = $_POST['experience'];
$income = $_POST['income'];

$sql = "INSERT into predict
(uname,category,subcategory,project_num,project_description,Experience,income) values
```

```
('".$uname."','".$category."','".$subcategory."','".$projects."','".$project_description."','".$experien
ce."','".$income."')";
if ($conn->query($sql) === TRUE) {
        header("Refresh:2; url=seller.html");
  echo "<html><script>alert('Average');</script>";

} else {
  echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();

?>
```

Database Connection Code

```
<?php
$servername = "127.0.0.1";
$username = "root";
$password = "";
$db = "website";

// Create connection
$conn = new mysqli($servername, $username, $password,$db);

// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

#echo "Connected successfully";

?>
```

CHAPTER 7

REFERENCES

[1]https://www.ijser.org/researchpaper/A-Systematic-Literature-Review-And-Case-Study-On-Influencing-Factor-And-Consequences-Of-Freelancing-In-Pakistan.pdf

[2]https://www.researchgate.net/publication/256039295_Freelance_Contracting_in_the_Digital_Age_Informality_Virtuality_and_Social_Ties

[3]http://ceur-ws.org/Vol-2077/paper6.pdf

[4]https://www.researchgate.net/publication/318980274_A_comparison_study_for_job_recommendation

[5]https://www.researchgate.net/publication/335771582_Explaining_and_exploring_job_recommendations_a_user-driven_approach_for_interacting_with_knowledge-based_job_recommender_systems

[6]https://www.researchgate.net/publication/271910489_Doing_database_design_with_MySQL

[7]https://www.researchgate.net/publication/320465713_A_Comparative_Study_of_Categorical_Variable_Encoding_Techniques_for_Neural_Network_Classifiers

[8]https://link.springer.com/chapter/10.1007/978-981-15-6198-6_18

[9]https://www.researchgate.net/publication/320171394_Research_Articles_in_Simplified_HTML_A_Web-first_format_for_HTML-based_scholarly_articles

[10]https://digitalcommons.wou.edu/cgi/viewcontent.cgi?article=1008&context=fac_pubs

[11]https://ieeexplore.ieee.org/document/7892476

[12]https://www.researchgate.net/publication/319164243_Natural_Language_Processing_State_of_The_Art_Current_Trends_and_Challenges

[13]https://garph.co.uk/ijarmss/july2013/15.pdf

[14]https://www.researchgate.net/publication/221621494_Support_Vector_Machines_Theory_and_Applications

[15]https://www.ijitee.org/wp-content/uploads/papers/v8i6s3/F10540486S319.pdf

[16]https://www.researchgate.net/publication/2948052_KNN_Model-Based_Approach_in_Classification

[17]https://www.researchgate.net/publication/259235118_Random_Forests_and_Decision_Trees

[18]https://www.researchgate.net/publication/342075482_Application_of_Logistic_Regression_in_Natural_Language_Processing

[19]https://www.researchgate.net/publication/220117802_A_comparison_of_the_bagging_and_the_boosting_methods_using_the_decision_trees_classifiers

[20]https://www.researchgate.net/publication/242579096_An_Introduction_to_Logistic_Regression_Analysis_and_Reporting

[21]https://www.researchgate.net/publication/338511842_Technological_Change_and_Innovation_as_Security_Threats

[22]https://ieeexplore.ieee.org/document/9411584