# UNIVERSITY ADMIT ELIGIBILITY PREDICTOR

# 1 INTRODUCTION

## 1.1 PROJECT OVERVIEW

The University Admit Eligibility Predictor we have Developed will be looking for the Probability of Admission for a Student in Universities. It predicts the Possibility using IBM Watson Machine Learning. It allows the Users to Provide Feedback for Further Development of the Project. It also resolves them to release their Stress of whether they would get the Admission or not.

## 1.2 PURPOSE

The Purpose of the Project is to make Students who willing to Join Universities get to know their Chances of Possibility in Admission across Universities. They will be Able to Know their Chances of University Admission. So they would be Confident on getting an Admission. The Project will be Able to Make Students free of the Fear of Failing to Get an Admission in the University and will be Them Focus on their Studies.

# 2 LITERATURE SURVEY

## 2.1 EXISTING PROBLEM

This paper proposes a new college admission prediction technique based on using two cascaded knowledge rules for achieving student's college admission with high performance fairly and accurately. The system analyzes student academic merits, background, student records, and the college admission criteria. Then, it predicts the likelihood of university college that a student may enter. For a pursuing graduate student, shortlisting the colleges could be an intense issue. College undergraduates frequently have an inclination to ponder over the chance that their profile suits the college requirements. Computer programs are exceptionally well trained and faster than humans in making decisions. Moreover, the cost of admission in a college is a lot,

making it very crucial for a student that their profile gets shortlisted for a university admission. Accurate prediction of college entrance examination(CEE) results is very important for the candidates to fill in the application and the relevant analysis of the CEE. At present, the prediction of CEE score is based on data statistics, probability model and some weighted combination models. In this paper, machine learning methods are used to carry out the admission lines of research and prediction. Specifically, in this paper the Adaboost algorithm is used to study and forecast, which belongs to ensemble learning. Finally, the result of this model is given, which is better than the current prediction.The engineering admission process is hectic and more than that is to find the college according to student preference i.e stream, location, university and seat type, etc. During the admission process the students always need to check the previous years cutoff of each college by visiting numerous websites & pdf lists which certainly consumes a lot of time. In our web application we will provide cutoff prediction of each college by the data-analysis of previous years cutoff, recommendation system for colleges listing according to student preferences, furthermore providing detailed comparison between institutions of their choice. The application is developed to bestow a personalized system so as to reduce time and ease the student's college selection process. Students applying for admissions to universities find it difficult to understand whether they have good chances of getting admission in a university or not. Keeping this in focus, we have used logistic regression techniques that have gained attention in the software engineering field for its ability to be used for predictions. This is a novel work on a university admissions predictor using which students can evaluate their competitiveness for getting admission at a university. This is developed by collecting real student data. The data is stored in a form of usable training data for the logistic regression classifier developed to make admissions predictions. We have collected the data from the Internet using a Selenium web scraper. The paper intensely discusses the methods, implementation and challenges faced in the process.

# 2.2 REFERENCES

***Paper 1***

*Authors:* Abdul Hamid M Ragab; Abdul Fatah S. Mashat; Ahmed M Khedra

*Year :* 2012

*Title:* HRSPCA : Hybrid Recommender System for Predicting College Admission

*Methodology:* This paper proposes a new college admission prediction technique based on using two cascaded knowledge rules for achieving student's college admission with high performance fairly and accurately. The system analyzes student academic merits, background, student records, and the college admission criteria. Then, it predicts the likelihood of university college that a student may enter.

*Advantage:* The system provides recommendations about which university colleges a student should be admitted to , taking into consideration not only student's scores but also other university qualified criteria into account. The HRSPCA system was validated using real students data.System experiments showed that the HRSPCA system performs substantially high performance due to allocating admission tasks between two cascading recommenders

*Disadvantage:* Out of 66 thousand applicants , Only 16 thousand were able to match the standard university or college standards.

***Paper 2***

*Authors:* Jayashree Katti, Jony Agarwal, Swapnil Bharata, Swati Shinde, Saral Mane, Vinod Biradar

*Year :* 2022

*Title:* University Admission Prediction using Google Vertex AI

*Methodology:* For a pursuing graduate student, shortlisting the colleges could be an intense issue. College undergraduates frequently have an inclination to ponder over the chance that their profile suits the college requirements. Computer programs are exceptionally well trained and faster than humans in making decisions. Moreover, the cost of admission in a college is a lot, making it very crucial for a student that their profile gets shortlisted for a university admission.

*Advantage:* A University prediction machine learning algorithm is very advantageous for college undergraduates to choose their dream university which also matches their

resume. The proposed method considers diverse variables related to the student and his score in various tests.

*Disadvantage:* The dataset includes LOR , GRE score , CGPA , TOFEL score , University Rating , SOP , etc. Based on all these criterias, the admission to a particular university of an undergraduate will be predicted.The Students may not be able to pay the Exam Fees of all the Above Exams.

***Paper 3***

*Authors:* Zhenru Wang , Yijie Shi

*Year :* 2016

*Title:* Prediction of the admission lines of college entrance examination based on machine learning

*Methodology:* Accurate prediction of college entrance examination(CEE) results is very important for the candidates to fill in the application and the relevant analysis of the CEE. At present, the prediction of CEE score is based on data statistics, probability model and some weighted combination models. In this paper, machine learning methods are used to carry out the admission lines of research and prediction. Specifically, in this paper the Adaboost algorithm is used to study and forecast, which belongs to ensemble learning. Finally, the result of this model is given, which is better than the current prediction.

*Advantage:* In this paper, machine learning methods are used to carry out the admission lines of research and prediction. Specifically, in this paper the Adaboost algorithm is used to study and forecast, which belongs to ensemble learning. Finally, the result of this model is given, which is better than the current prediction.

*Disadvantage:* Of course, the establishment of the model is not very perfect because of the in-exhaustive data. And there are still a lot of things to be improved. And in the aspect of feature selection, we only forecast the college entrance examination of Sichuan

province. If we get more data in the future, we can also do university admission line forecasts. It is also a very significant work.

***Paper 4***

*Authors:* Abdul Majeed Inamdar, Tanmay Mhatre, Pravin Nadar, Supriya Joshi

*Year :* 2022

*Title:* Personalized College Recommender and Cutoff Predictor for Direct Second Year Engineering

*Methodology:* The engineering admission process is hectic and more than that is to find the college according to student preference i.e stream, location, university and seat type, etc. During the admission process the students always need to check the previous years cutoff of each college by visiting numerous websites & pdf lists which certainly consumes a lot of time. In our web application we will provide cutoff prediction of each college by the data-analysis of previous years cutoff, recommendation system for colleges listing according to student preferences, furthermore providing detailed comparison between institutions of their choice. The application is developed to bestow a personalized system so as to reduce time and ease the student's college selection process.

*Advantage:* It is very clear on the Goal of the Project and is very productive in a large set available for them.

*Disadvantage:* It doesn't work well for smaller sets while training. Such larger Sets are not available on the Web.

***Paper 5***

*Authors:* Haseeba Fathiya, Lipsa Sadath

*Year :* 2021

*Title:* University Admissions Predictor Using Logistic Regression

*Methodology:* Students applying for admissions to universities find it difficult to understand whether they have good chances of getting admission in a university or not. Keeping this in focus, we have used logistic regression techniques that have gained attention in the software engineering field for its ability to be used for predictions. This is a novel work on a university admissions predictor using which students can evaluate their competitiveness for getting admission at a university. This is developed by collecting real student data. The data is stored in a form of usable training data for the logistic regression classifier developed to make admissions predictions. We have collected the data from the Internet using a Selenium web scraper. The paper intensely discusses the methods, implementation and challenges faced in the process.

*Advantage:* The web scraper saves a lot of time and reduces labor costs, and the admissions predictor can be a useful tool to students trying to narrow down their university choices

*Disadvantage:* Many new features can be incorporated to improve the application. The model selected for each university can be tailored to produce the best results for the data available. Experimentation can also be done using other classification algorithms which could improve the accuracy. Other features can be included along with test scores and GPA to produce more accurate results.

| S.No | Author | Title of the Paper | Methodology | Pros (Advantage) | Cons (Disadvantage) |
|---|---|---|---|---|---|
| 1. | Abdul Hamid M Ragab; Abdul Fatah S. Mashat; Ahmed M | HRSPCA : Hybrid Recommender System for Predicting College Admission | This paper proposes a new college admission prediction technique based on using two cascaded knowledge rules for achieving | The system provides recommendations about which university colleges a student should be admitted to , taking into | Out of 66 thousand applicants , Only 16 thousand were able to match the standard university or college |

| | | | | | |
|---|---|---|---|---|---|
| | Khedra (2012) **(IEEE paper 1)** | | student's college admission with high performance fairly and accurately. The system analyzes student academic merits, background, student records, and the college admission criteria. Then, it predicts the likelihood of university college that a student may enter. | consideration not only student's scores but also other university qualified criteria into account. The HRSPCA system was validated using real students data.System experiments showed that the HRSPCA system performs substantially high performance due to allocating admission tasks between two cascading recommenders | standards. |
| 2. | Jayashree Katti, Jony Agarwal, Swapnil Bharata, Swati Shinde, Saral Mane, Vinod Biradar (2022) | University Admission Prediction using Google Vertex AI | For a pursuing graduate student, shortlisting the colleges could be an intense issue. College undergraduates frequently have an inclination to ponder over the chance that their profile suits the college | A University prediction machine learning algorithm is very advantageous for college undergraduates to choose their dream university which also matches their resume. The proposed method | The dataset includes LOR , GRE score , CGPA , TOFEL score , University Rating , SOP , etc. Based on all these criterias, the admission to a particular university of an |

| | | | | | |
|---|---|---|---|---|---|
| | **(IEEE paper 3)** | | requirements. Computer programs are exceptionally well trained and faster than humans in making decisions. Moreover, the cost of admission in a college is a lot, making it very crucial for a student that their profile gets shortlisted for a university admission. | considers diverse variables related to the student and his score in various tests. | undergraduate will be predicted.The Students may not be able to pay the Exam Fees of all the Above Exams. |
| 3. | Zhenru Wang , Yijie Shi (2016) **(IEEE Paper 4)** | Prediction of the admission lines of college entrance examination based on machine learning | Accurate prediction of college entrance examination results is very important for the candidates to fill in the application and the relevant analysis of the CEE. At present, the prediction of CEE score is based on data statistics, probability model and some weighted combination | In this paper, machine learning methods are used to carry out the admission lines of research and prediction. Specifically, in this paper the Adaboost algorithm is used to study and forecast, which belongs to ensemble learning. Finally, the result of this model is given, which is better | Of course, the establishment of the model is not very perfect because of the in-exhaustive data. And there are still a lot of things to be improved. And in the aspect of feature selection, we only forecast the college entrance examination of Sichuan province. If we |

| | | | | | |
|---|---|---|---|---|---|
| | | | models. Machine learning methods are used to carry out the admission lines of research and prediction.In this paper the Adaboost algorithm is used to study and forecast, which belongs to ensemble learning. | than the current prediction. | get more data in the future, we can also do university admission line forecasts. It is also a very significant work. |
| 4. | Abdul Majeed Inamdar, Tanmay Mhatre, Pravin Nadar, Supriya Joshi<br><br>(2022) **(IEEE Paper 5)** | Personalized College Recommender and Cutoff Predictor for Direct Second Year Engineering | The engineering admission process is hectic and more than that is to find the college according to student preference i.e stream, location, university and seat type, etc. During the admission process the students always need to check the previous years cutoff of each college by visiting numerous websites & pdf lists which | It is very clear on the Goal of the Project and is very productive in a large set available for them. | It doesn't work well for smaller sets while training. Such larger Sets are not available on the Web. |

| | | | | | |
|---|---|---|---|---|---|
| | | | certainly consumes a lot of time. In our web application we will provide cutoff prediction of each college by the data-analysis of previous years cutoff, recommendation system for colleges listing according to student preferences, furthermore providing detailed comparisons between institutions of their choice. The application is developed to bestow a personalized system so as to reduce time and ease the student's college selection process. | | |
| 5. | Haseeba Fathiya, Lipsa Sadath (2021) | University Admissions Predictor Using Logistic | Students applying for admissions to universities find it difficult to understand | The web scraper saves a lot of time and reduces labor costs, and the admissions | Many new features can be incorporated to improve the application. The |

| | | | | | |
|---|---|---|---|---|---|
| | **(IEEE Paper 6)** | Regression | whether they have good chances of getting admission in a university or not. Keeping this in focus, we have used logistic regression techniques that have gained attention in the software engineering field for its ability to be used for predictions. This is a novel work on a university admissions predictor using which students can evaluate their competitiveness for getting admission at a university. This is developed by collecting real student data. The data is stored in a form of usable training data for the logistic regression classifier developed to make admissions | predictor can be a useful tool to students trying to narrow down their university choices | model selected for each university can be tailored to produce the best results for the data available. Experimentation can also be done using other classification algorithms which could improve the accuracy. Other features can be included along with test scores and GPA to produce more accurate results. |

| | | | | | |
|---|---|---|---|---|---|
| | | | predictions. We have collected the data from the Internet using a Selenium web scraper. The paper intensely discusses the methods, implementation and challenges faced in the process. | | |

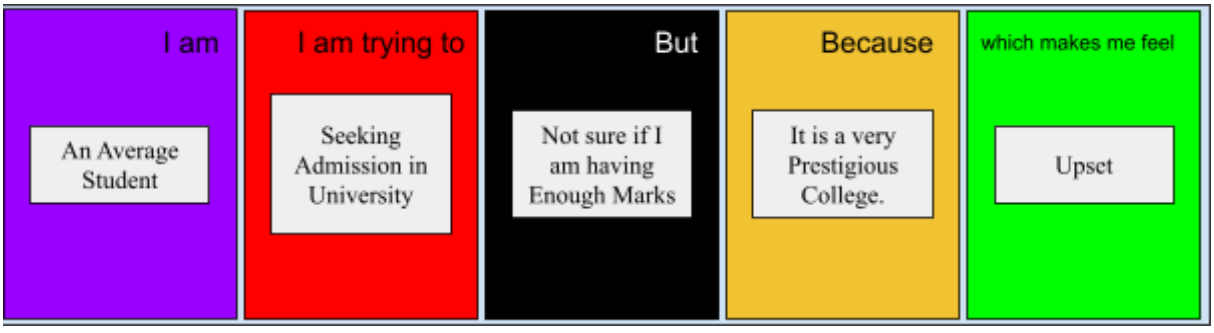
## 2.3 Problem Statement Definition

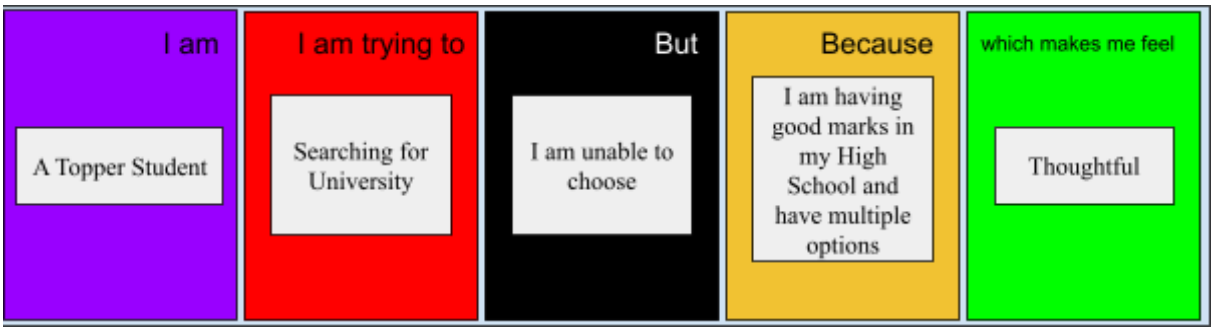
Nowadays Students after Completing their Board Exams are having High Expectations in their Higher Studies. But During their Admission, They are less likely to get the University they wished they wanted to get. So We are creating a University Admit Eligibility Predictor where the Students will be able to check the Possibility of their admission in the University they wished to Join. We Train the Machine Learning Model with the Admission Details of various Universities. We also provide Details of the Fees Collected by the Universities to give them an idea on the Expenses to be allocated for the Higher Education Purpose. The Students can also be Confident that the University they are expecting accepts students with the Mark he/she is having. So They will not have any Sad Feelings when they don't get the University they are Expecting.

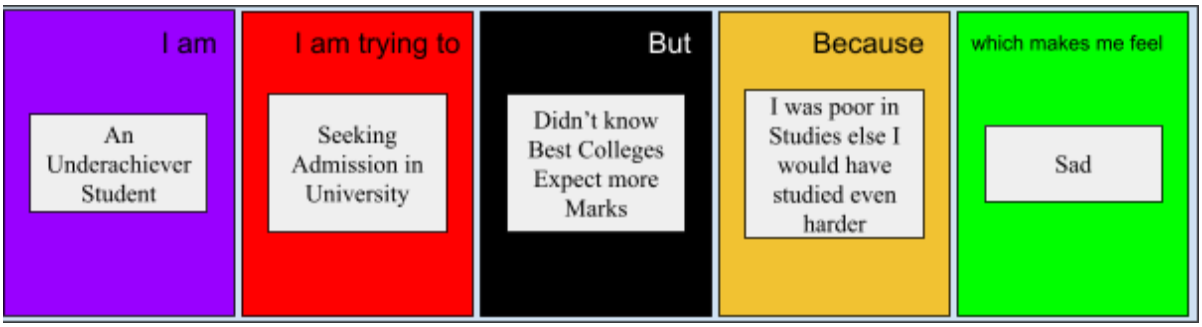
We will be conducting an English Test to make sure that they will be able to make up for the Communication gap in case they are opting for a University outside of their Native Place.

Through this Project, We will be able to do a University Admit Eligibility Predictor that will be able to predict the Possibility of a Student's Admission in a Specific University based on their marks.

| | |
|---|---|
| Who does the Problem affect? | Students Seeking University Admission |
| What are the Boundaries of the Problem | Students who want to ensure they are Eligible for Specific University |
| What is the Issue? | Students are not able to see Every Universities Cutoff to Check if they are Eligible for the Respective University. |
| When does the Issue Occur? | Mostly during the Month when Result is Announced |
| Where does the Issue Occur? | It occurs in all Educational Institutions |
| Why is it Important that we fix the Problem? | Students get Depressed when they don't get the University they are Expecting and are not able to concentrate on further Studies and perform poorly. |
| What Solution solves this Issue? | A Model Trained well enough to predict the Possibilities of a Student getting Admission in a Specific University |
| What Methodology is used to solve this Issue? | Machine Learning is used for training the Model to be able to Predict the Possibility of a Student being able to join a Specific University. |

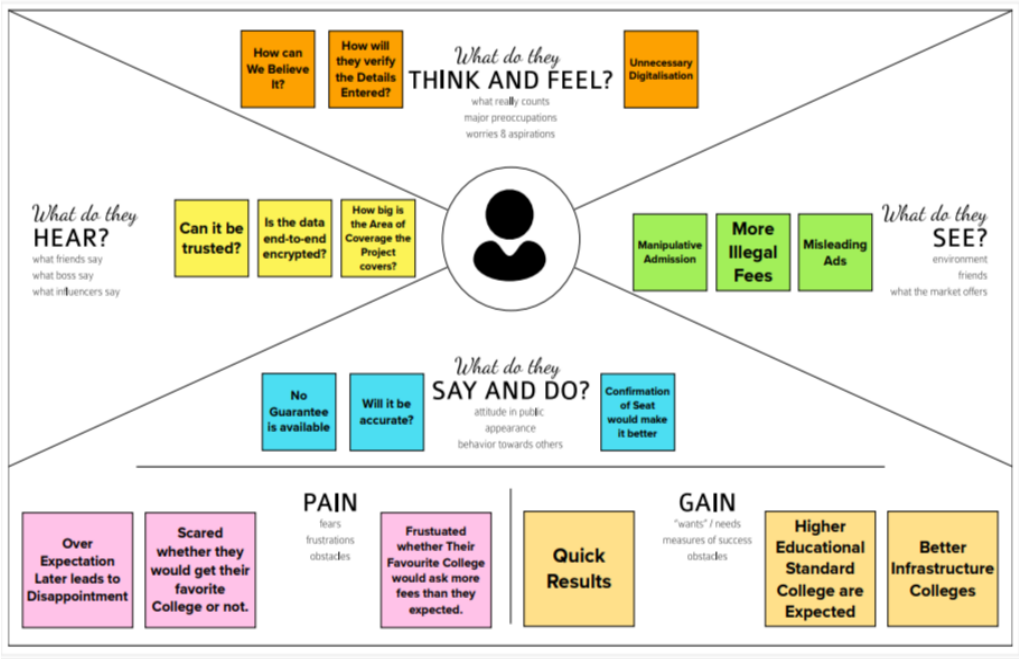




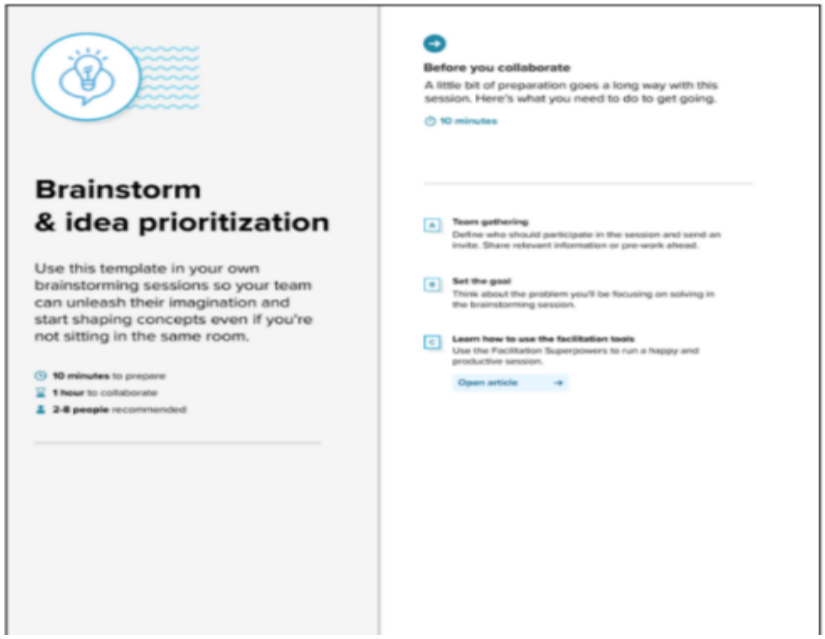

# 3. IDEATION & PROPOSED SOLUTION

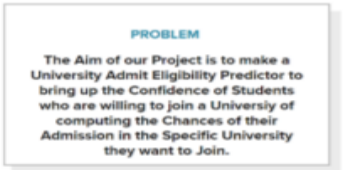
## 3.1 Empathy Map Canvas



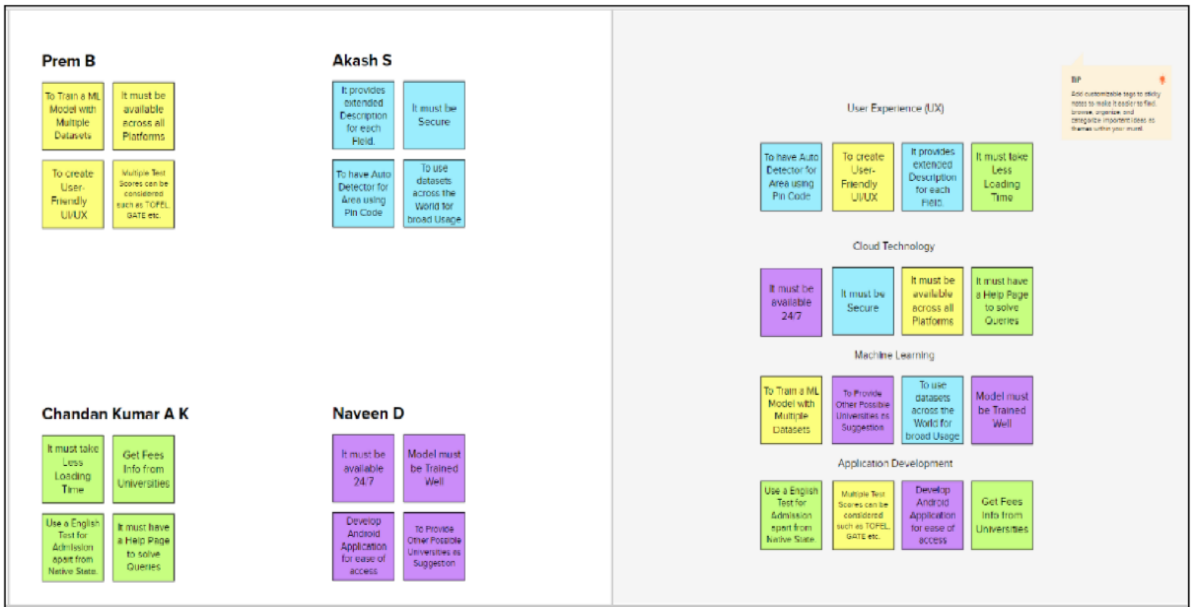
# 3.2 Ideation & Brainstorming

**STEP 1 : TEAM GATHERING , COLLABORATING AND SELECT THE PROBLEM STATEMENT**

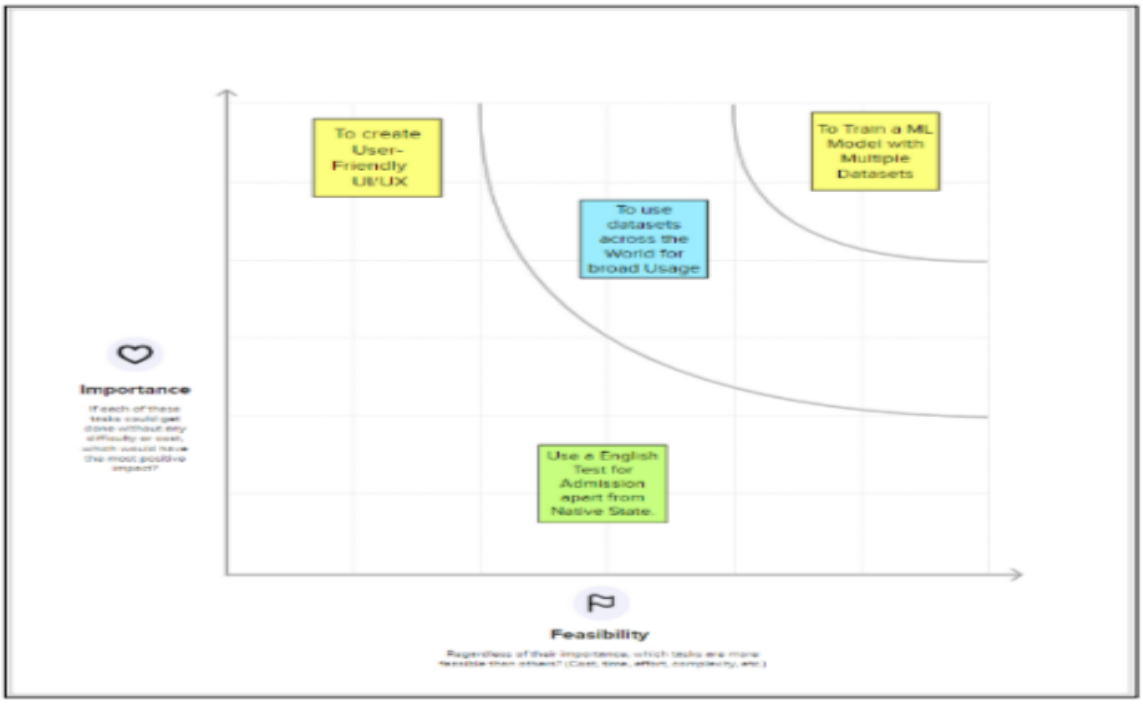



**STEP 2 : BRAINSTORMING , IDEA LISTING AND GROUPING**



**STEP 3 : IDEA PRIORITIZATION**



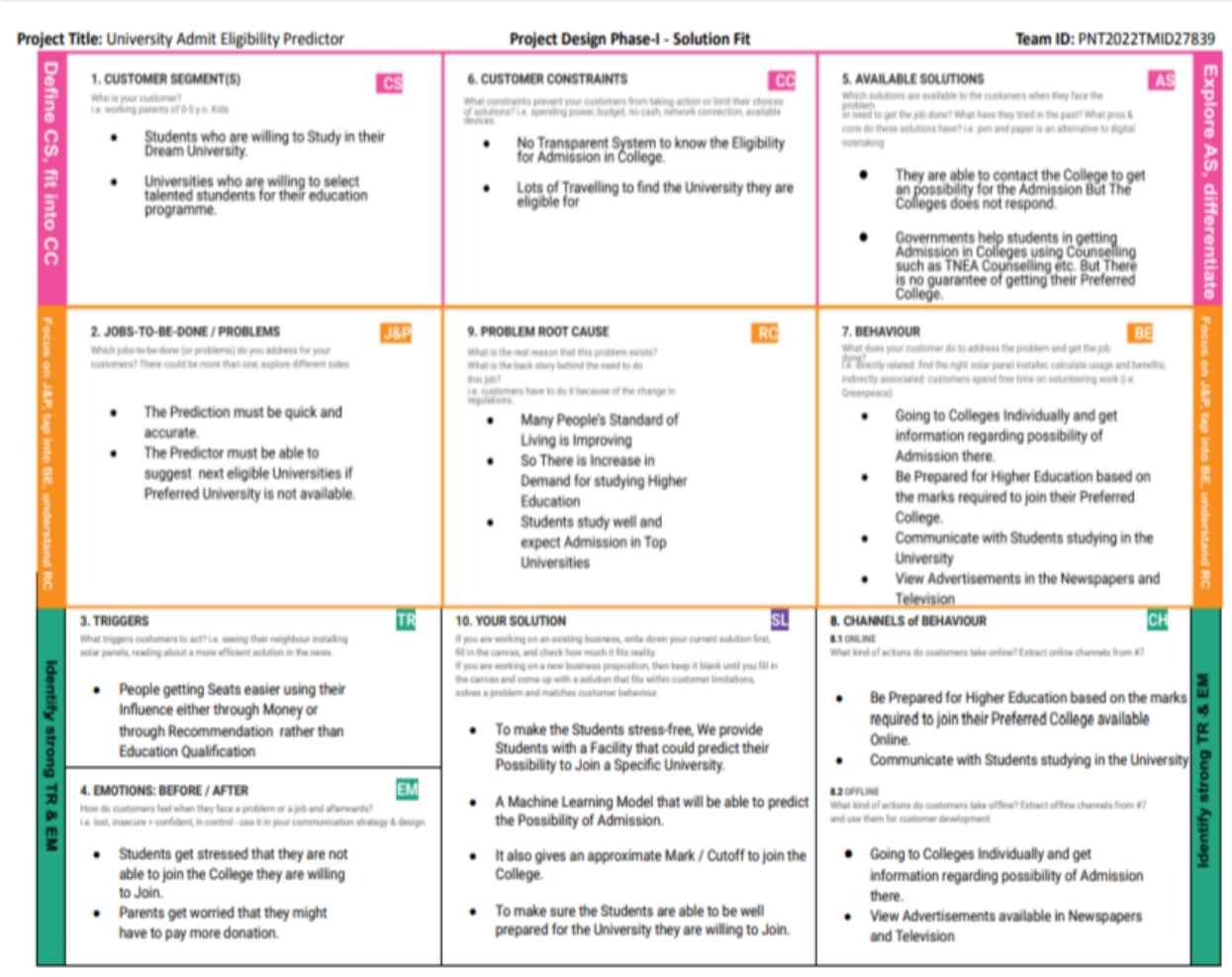

# 3.3 Proposed Solution

| S. No | PARAMETER | DESCRIPTION |
|---|---|---|
| 1. | Problem Statement (Problem to be Solved) | Students now-a-days are very stressed that they are not able to get the College they wanted to Join. So They are not able to concentrate on their Studies<br>well as they used to. They are a bit disappointed when they don't get the University they preferred.<br>They are asked for more Donations if not met with the Criteria. |

| | | |
|---|---|---|
| 2. | Idea / Solution Description | To make the Students stress-free, We are developing a University Admit Eligibility Predictor to provide Students with a Facility that could predict their Possibility to Join a Specific University. |
| 3. | Novelty / Uniqueness | The Predictor not only just Predicts the Possibility of Admission but also Provides similar Admissible Universities as Suggestions for a backup for the Students where they join in case the University they preferred has a higher Criteria. |
| 4. | Social Impact / Customer Satisfaction | Through this , we are able to reduce the stress developed among the Students during the Admission Time. We are able to achieve a Transparent System that is open to all and anyone is able to view the Possibility of Admission without Restrictions of any kind. It prevents the possibility of Parents disapproving of the Children's Higher Education due to higher costs. |
| 5. | Business Model (Revenue Model) | The Predictor also displays the Average amount of Fees Collected by the College per annum which helps the Students ensure they will be able to afford the University Admission. Through this Predictor, we are able to reduce the Amount Spent by Students during the Admission Period and are able to save a considerable amount of money through it. |

| | | |
|---|---|---|
| 6. | Scalability of the Solution | The Project is initially developed for a Specific University but it is scalable to National or International Universities also. It could later be upgraded to compute the Donation requested by the University for the Cutoff the Student has. The Project can also show Currency Conversion for Admissions in Universities Abroad. These kinds of Upgrade are Scalable in this Project. |

# 3.4 Problem Solution fit



# 4 REQUIREMENT ANALYSIS

## 4.1 Functional requirement

### FUNCTIONAL REQUIREMENTS:

| FR No. | FUNCTIONAL REQUIREMENT (EPIC) | SUB REQUIREMENT (STORY / SUB-TASK) |
|---|---|---|
| 1.) | Home Page | It provides information regarding the Predictor and how it works and will have a Sitemap to the Other Pages. |
| 2.) | User Input | The User will be providing the Score he had got in his Academics and his personal Details which will be used to predict his chances of Admission |
| 3.) | Score Analysis | The Scores are used to predict the Possibilities of Admission in a well trained Model deployed in the Cloud. |
| 4.) | Score Result | The Results ie the Possibility of Admission is displayed to the User and Also Other Possible Universities where he could Join are Also Suggested. |
| 5.) | University Page | The Details of Universities that are available are displayed in this Page. So the Students are able to view the Stats of the University. |
| 6.) | Feedback | The User Provides his Feedback where he can provide certain suggestions to be Implemented or the Part of the Project where it is Good. |

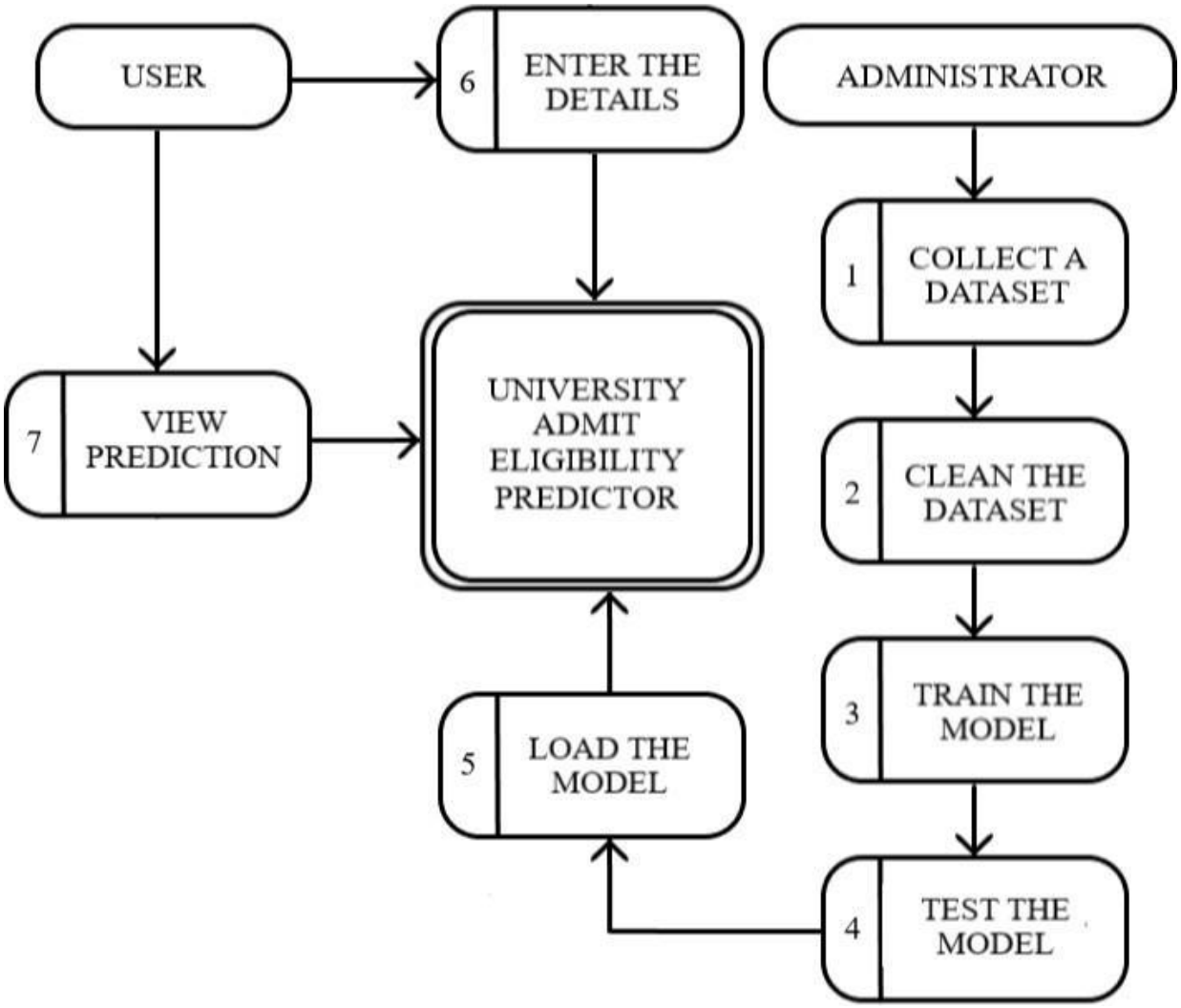
## 4.2 Non-Functional requirements

### NON FUNCTIONAL REQUIREMENTS:

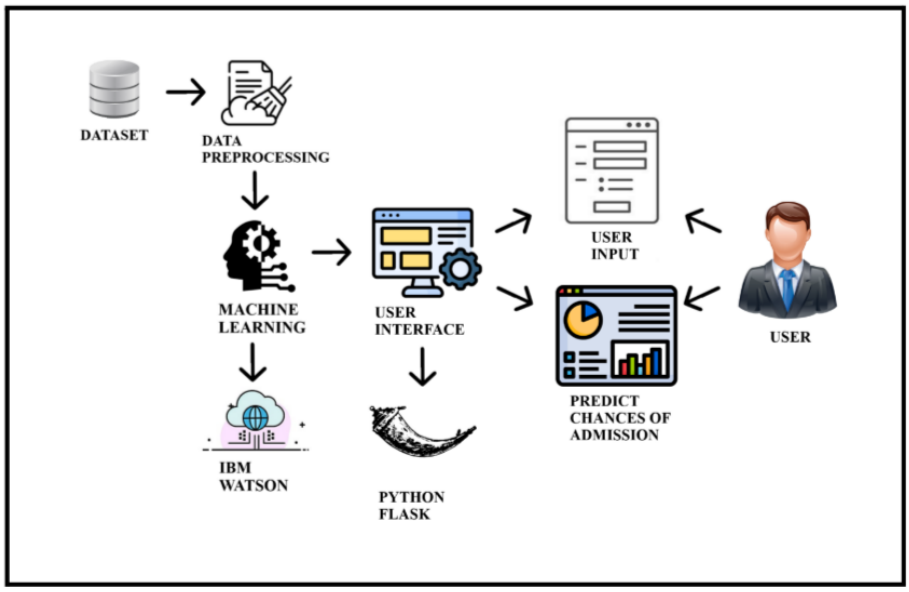
| NFR No. | NON-FUNCTIONAL REQUIREMENT | DESCRIPTION |
|---|---|---|
| 1.) | **Usability** | It has high usability because most of the Students will be willing to join Higher Education. It will be user-friendly and very minimalistic information is required. |
| 2.) | **Security** | The information provided by users are provided with maximum security and complete Confidentiality. None will be able to view the Information provided by the Information. |
| 3.) | **Reliability** | The Prediction is highly Reliable because The Model is Well Trained with Data of Universities across the world and with Better Accuracy. |
| 4.) | **Performance** | The Performance of the Admission Predictor is because it has a very less loading Time. It has very few Graphics which also takes only a little space. It is highly accurate in predicting the Admission of the Student. |
| 5.) | **Availability** | It is made available to the Users at any Time without any Restriction 24*7 Because The Model is Deployed in IBM Cloud , which is made to work under any such Circumstances. |
| 6.) | **Scalability** | Any Number of Users can Access the Predictor without any Issue Because The Model is Deployed in IBM Cloud , which is made to work under any such Circumstances. |

# 5 PROJECT DESIGN

## 5.1 Data Flow Diagrams



## 5.2 Solution & Technical Architecture




## 5.3 User Stories

| USER TYPE | FUNCTIONAL REQUIREMENT (EPIC) | USER STORY NUMBER | USER STORY / TASK | ACCEPTANCE CRITERIA | PRIORITY | RELEASE |
|---|---|---|---|---|---|---|
| Customer (Student) | Home | USN - 1 | As a User, I will be able to view the Details of the Predictor. | The Information regarding the Predictor is available in Detail. | Low | Sprint - 3 |
| | Details Form | USN - 2 | I have to fill the Form with Academic Details and provide information to the Predictor. | I am able to Fill the Forms and am notified if any input is given abnormally. | Medium | Sprint - 2 |
| | Analysis | USN - 3 | The Form Results are sent to the Model to predict the Possibilities and Results are Obtained | The Form Input is Sent to Model Securely and The Results are Fetched Successfully. | High | Sprint -1 |
| | Result | USN - 4 | I have to be able to view the Results of the Prediction | I am able to see the chances of Admission in the University. | Medium | Sprint - 2 |
| | | USN - 5 | I have to be able to view Alternate Universities if the Chances of My Preferred University is very less. | I am able to see Alternate Universities if Chances of My Preferred University are less. | Medium | Sprint - 3 |
| | Feedback | USN - 6 | After Completing my Prediction, I have to Share a Feedback to the Prediction for further Improvements | I have shared the Feedback to the Administrator. | Low | Sprint - 4 |
| | Share to Friends | USN - 7 | I will be able to share the Predictor to Others. | I am able to Share it. | Low | Sprint - 4 |
| Administrator | Model Generation | USN - 7 | I have to train a Model that is able to predict the Chances of Admission Accurately | The Model is well trained and Results are accurate. | High | Sprint - 1 |
| | Model Updation | USN - 8 | I have to change the Existing Model with a new Model as it gets Outdated as time passes. | The Model is now Changeable at any Time. | Medium | Sprint - 1 |
| Technical Support | Technical Improvement | USN - 10 | I should be able to view Feedbacks and Try to resolve their Query. | The Feedbacks are generated as a Report and are displayed. | Low | Sprint - 4 |

# 6 PROJECT PLANNING & SCHEDULING

## 6.1 Sprint Planning & Estimation

**PRODUCT BACKLOG , SPRINT SCHEDULE AND ESTIMATION:**

| RELEASE | FUNCTIONAL REQUIREMENT (EPIC) | USER STORY NUMBER | USER STORY / TASK | STORY POINTS | PRIORITY | TEAM MEMBERS |
|---|---|---|---|---|---|---|
| Sprint - 3 | Home | USN - 1 | As a User , I will be able to view the Details of the Predictor. | 5 | Low | Chandan |
| Sprint - 2 | Details Form | USN - 2 | I have to fill the Form with Academic Details and provide information to the Predictor. | 6 | Medium | Prem , Akash |
| Sprint -1 | Analysis | USN - 3 | The Form Results are sent to the Model to predict the Possibilities and Results are Obtained | 9 | High | Prem, Naveen |
| Sprint - 2 | Result | USN - 4 | I have to be able to view the Results of the Prediction | 8 | Medium | Chandan |
| Sprint - 3 | | USN - 5 | I have to be able to view Alternate Universities if the Chances of My Preferred University is very less. | 10 | Medium | Prem , Akash |
| Sprint - 4 | Feedback | USN - 6 | After Completing my Prediction, I have to Share a Feedback to the Prediction for further Improvements | 7 | Low | Naveen |
| Sprint - 4 | Share to Friends | USN - 7 | I will be able to share the Predictor to Others. | 8 | Low | Prem |
| Sprint - 1 | Model Generation | USN - 8 | I have to train a Model that is able to predict the Chances of Admission Accurately | 13 | High | Prem , Chandan |
| Sprint - 1 | Model Updation | USN - 9 | I have to change the Existing Model with a new Model as it gets Outdated as time passes. | 8 | Medium | Naveen , Akash |
| Sprint - 4 | Technical Improvement | USN - 10 | I should be able to view Feedbacks and Try to resolve their Query. | 6 | Low | Akash |

## 6.2 Sprint Delivery Schedule

**PROJECT TRACKER , VELOCITY & BURNDOWN CHART :**

| SPRINT | TOTAL STORY POINTS | DURATION | SPRINT START DATE | SPRINT END DATE (PLANNED) | STORY POINTS COMPLETED (AS PER PLANNED DATE) | SPRINT RELEASE DATE (ACTUAL) |
|---|---|---|---|---|---|---|
| SPRINT - 1 | 30 | 6 Days | 24th Oct 2022 | 29th Oct 2022 | 30 | 29th Oct 2022 |
| SPRINT - 2 | 14 | 6 Days | 31th Oct 2022 | 5th Nov 2022 | 14 | 5th Nov 2022 |
| SPRINT - 3 | 15 | 6 Days | 7th Nov 2022 | 12th Nov 2022 | 15 | 12th Nov 2022 |
| SPRINT - 4 | 21 | 6 Days | 14th Nov 2022 | 19th Nov 2022 | 21 | 19th Nov 2022 |

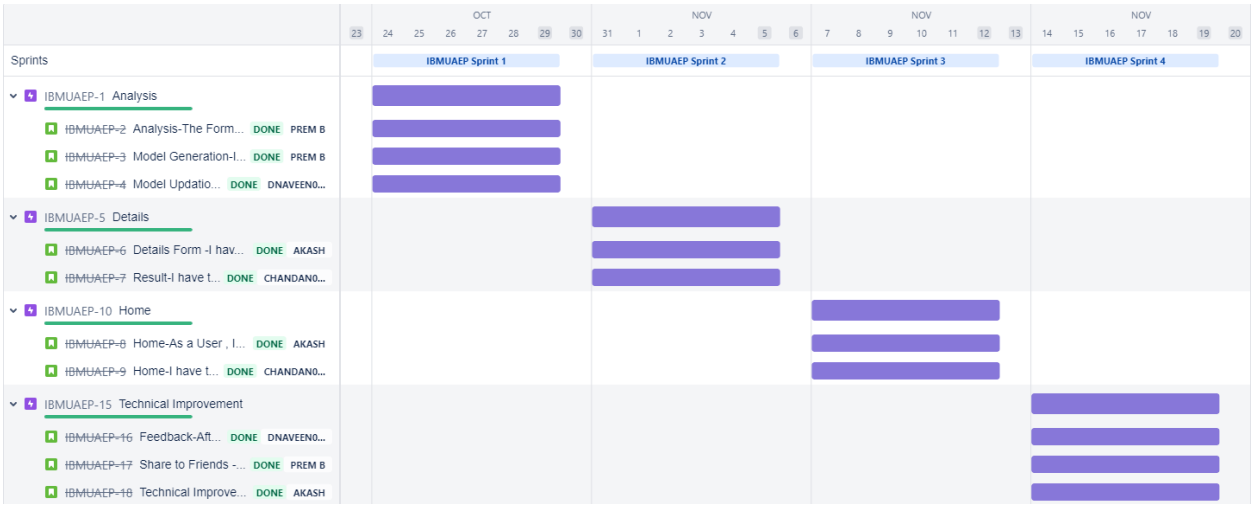
**VELOCITY :**

$$AV = \frac{sprint\ duration}{velocity}$$

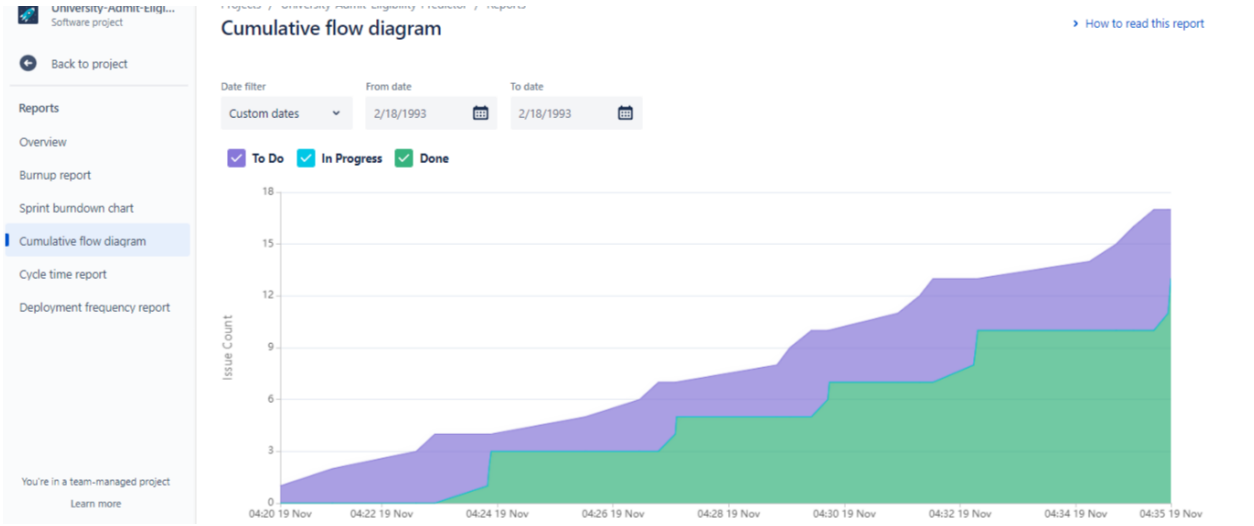
| SPRINT | TOTAL STORY POINTS | DURATION | AVERAGE VELOCITY |
|---|---|---|---|
| SPRINT -1 | 30 | 6 Days | $\frac{30}{6} = 5$ |
| SPRINT - 2 | 14 | 6 Days | $\frac{14}{6} = 2.33$ |
| SPRINT - 3 | 15 | 6 Days | $\frac{15}{6} = 2.5$ |
| SPRINT - 4 | 21 | 6 Days | $\frac{21}{6} = 3.5$ |
| **OVERALL** | **80** | **24 Days** | $\frac{80}{24} = \mathbf{3.33}$ |

# 6.3 REPORTS FROM JIRA

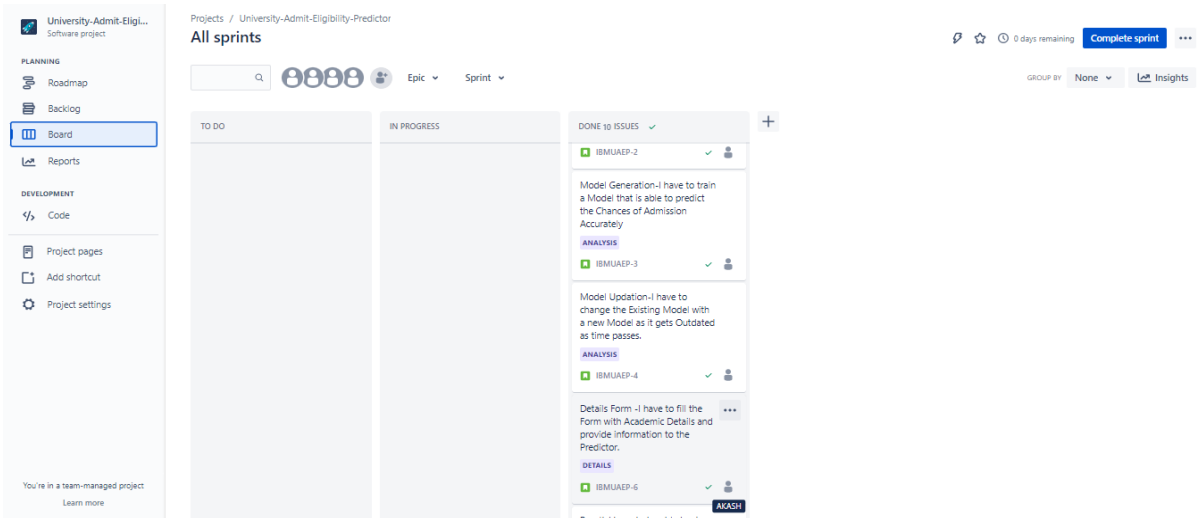
## Road map




## Cumulative Flow Diagram



# Scrum Board




# 7 CODING & SOLUTIONING

## 7.1 CLIENT SIDE APPLICATION

```
from flask import Flask, render_template, request
from flask_cors import CORS
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "PLACE THE API KEY HERE"

token_response = requests.post("https://iam.cloud.ibm.com/identity/token",data = {
    "apikey":API_KEY , "grant_type":"urn:ibm:params:oauth:grant-type:apikey"
})

mltoken = token_response.json()["access_token"]

header = {"Content-Type" : "application/json" , "Authorization" : "Bearer " + mltoken}


app = Flask(__name__,static_url_path='')
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
```

```
# No cacheing at all for API endpoints.
@app.after_request
def add_header(response):
    # response.cache_control.no_store = True
    if 'Cache-Control' not in response.headers:
        response.headers['Cache-Control'] = 'no-store'
    return response

CORS(app)

@app.route('/',methods=['GET','POST'])
def sendHomePage():
    if(request.method == "GET"):
        return render_template('index.html' , notification = 'Welcome!')
    else:
        return render_template('index.html' , notification = 'You have given ' +
request.form['rating'] + ' Stars Feedback')

@app.route('/predict',methods=['POST'])
def PredictPossibility():
    GREScore = float(request.form['GREScore'])
    TOEFLScore = float(request.form['TOEFLScore'])
    UnivRating = float(request.form['UnivRating'])
    SOP = float(request.form['SOP'])
    LOR = float(request.form['LOR'])
    CGPA = float(request.form['CGPA'])
    Research = 0
    if('Research' in request.form):
        Research = 1
    X = [[GREScore , TOEFLScore , UnivRating , SOP , LOR , CGPA , Research ]]
    print(X)
    # NOTE: manually define and pass the array(s) of values to be scored in the next line
    payload_scoring = {"input_data": [{"fields":
[["GREScore","TOEFLScore","UnivRating","SOP","LOR","CGPA","Research"]], "values":
X}]}
    response_scoring = requests.post('https://us-
```

```
south.ml.cloud.ibm.com/ml/v4/deployments/1b254abd-eaec-4f59-9e86-
a3674f3eea4c/predictions?version=2022-11-15', json=payload_scoring,
    headers=header)
    print("Scoring response")
    print(response_scoring.json())
    probability = int(round((response_scoring.json()["predictions"][0]["values"][0][0]),2)*100)
    print(probability)
    prob_comment = ""
    color_scheme = ""
    if(probability > 100):
        probability = 100
    elif(probability < 0):
        probability = 0
    if(probability < 50):
        prob_comment = "The Chances of Getting an Admission is less likely"
        color_scheme = 'darkorange'
    elif(probability < 70):
        prob_comment = "There is a slight Chance of Possibility."
        color_scheme = 'yellow'
    else:
        prob_comment = "There is High Chances of Possibility"
        color_scheme = 'lawngreen'
    return
render_template('predict.html',predict=probability,comment=prob_comment,color_scheme=co
lor_scheme)

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

# 7.2 SERVER SIDE MACHINE LEARNING

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
get_ipython().run_line_magic('matplotlib', 'inline')
df = pd.read_csv('dataset/Admission_Predict.csv')
df.head()
```

```
df.drop(["Serial No."],axis = 1,inplace = True)
df.head()
df.describe()
df.info()
df.isnull().sum()
sns.distplot(df['TOEFL Score'])
plt.pie(df['University Rating'].value_counts(),[0.05,0.05,0.05,0.05,0.05],df['University
Rating'].value_counts().index,autopct='%1.1f%%')
plt.title('University Rating')
plt.plot()
sns.barplot(df.Research.value_counts().index,df.Research.value_counts())
sns.lineplot(df['GRE Score'],df['Chance of Admit'])
def countplot_2(x,hue,title=None,figsize=(12,10)):
    plt.figure(figsize=figsize)
    sns.countplot(data = df[[x,hue]],x=x,hue=hue)
    plt.title(title)
    plt.plot()
countplot_2('Research','University Rating','University based Research Ability')
sns.pairplot(df)
df.corr()
plt.figure(figsize=(12,10))
sns.heatmap(df.corr(),annot=True)
plt.show()
for i in df.columns:
    sns.boxplot(df[i])
    plt.show()
for i in df.columns:
    Q1 = df[i].quantile(0.25)
    Q3 = df[i].quantile(0.75)
    IQR = Q3-Q1
    upper_limit = Q3 + (1.5*IQR)
    lower_limit = Q1 - (1.5*IQR)
    df[i] = np.where(df[i]>=upper_limit,Q3 + (1.5*IQR),df[i])
    df[i] = np.where(df[i]<=lower_limit,Q1 - (1.5*IQR),df[i])
for i in  df.columns:
    sns.boxplot(df[i])
    plt.show()
```

```
X = df.drop(columns=['Chance of Admit'])
X
Y = pd.DataFrame(df['Chance of Admit'])
Y
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
X_Scaled = pd.DataFrame(scale.fit_transform(X),columns = X.columns)
target_scale = MinMaxScaler()
Y_Scaled = pd.DataFrame(target_scale.fit_transform(Y),columns = Y.columns)
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size = 0.2,random_state=90)
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train,y_train)
y_test.to_numpy().flatten()
y_predict = model.predict(x_test)
pd.DataFrame({'Actual':y_test.to_numpy().flatten(),'Predicted':y_predict.flatten()})
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
print(r2_score(y_test,y_predict))
print(mean_absolute_error(y_test,y_predict))
print(mean_squared_error(y_test,y_predict))
print(mean_squared_error(y_test,y_predict,squared=False))
import joblib
joblib.dump(model,r'model/Admission_Predictor.pkl')
```

# 8 TESTING

## 8.1 Test Cases

- Check Whether The Form is Visible
- Check Whether The Submit Button is Clickable
- Check Whether the Form Data gets Submitted Successfully to Result Page
- Check Whether Displayed Predicted Precentage of Admission
- Check Whether If Values are Given Outside of Range It will provide Error Message

Regarding the Range of that Particular Input

Check Whether the Notification Bar Displays 'Welcome' on Entry to the Website.

Check Whether the Feedback Form is Displayed

Check Whether the Emoji are Selectable in Feedback Form

Check Whether Submit Button is Clickable

Check Whether the Notification Bar Displays the Feedback Given.

## 8.2 User Acceptance Testing

This Report Shows the Number of Bugs resolved or closed at each Severity Level and How they were Resolved

| Resolution | Severity 1 | Severity 2 | Severity 3 | Severity 4 | Subtotal |
|---|---|---|---|---|---|
| By Design | 7 | 4 | 1 | 3 | 15 |
| Duplicate | 3 | 2 | 0 | 1 | 6 |
| External | 1 | 0 | 4 | 2 | 7 |
| Fixed | 9 | 4 | 3 | 5 | 21 |
| Not Reproduced | 1 | 0 | 1 | 2 | 4 |
| Skipped | 0 | 0 | 1 | 2 | 3 |
| Won't Fix | 1 | 3 | 1 | 0 | 5 |
| Total | 22 | 13 | 11 | 15 | 61 |

This Report Shows the Number of Test Cases that have Passed , Failed and untested.

| Section | Test Cases | Untested | Fail | Pass |
|---|---|---|---|---|
| Print Engine | 2 | 0 | 0 | 2 |
| Client Application | 32 | 0 | 0 | 32 |
| Security | 3 | 0 | 0 | 3 |
| Outsource Shipping | 2 | 0 | 0 | 2 |
| Exception Reporting | 10 | 0 | 0 | 10 |
| Final Report Output | 7 | 0 | 0 | 7 |
| Version Control | 5 | 0 | 0 | 5 |

# 9 RESULTS

## 9.1 Performance Metrics

### EVALUATE THE PERFORMANCE USING METRICS

#### R2 Score

In [14]:
```
from sklearn.metrics import r2_score
print(r2_score(y_test,y_predict))
```

```
0.8724323452611149
```

#### Mean Absolute Error

In [15]:
```
from sklearn.metrics import mean_absolute_error
print(mean_absolute_error(y_test,y_predict))
```

```
0.039251015019866026
```

#### Mean Sqaured Error

In [16]:
```
from sklearn.metrics import mean_squared_error
print(mean_squared_error(y_test,y_predict))
```

```
0.0025295186149925943
```

#### Root Mean Squared Error

In [17]:
```
print(mean_squared_error(y_test,y_predict,squared=False))
```

```
0.05029431990784441
```

### MODEL TUNING

#### KFOLD VALIDATION METHOD

In [18]:
```
from sklearn.model_selection import KFold,cross_val_score
kfold = KFold(n_splits=10, shuffle=True , random_state=100)
model_kfold = LinearRegression()
results_kfold = cross_val_score(model_kfold, X, Y, cv=kfold)
print("Accuracy: %.2f%%" % (results_kfold.mean()*100.0))
```

```
Accuracy: 80.22%
```

# 10 ADVANTAGES & DISADVANTAGES

## ADVANTAGES

It is very Quick.

The Results are Accurate.

It is Open Sourced.

No External Payment Required.

Secure. No need to Provide Extra Information such as Login Email ,Password etc.

## DISADVANTAGES

It is possible target to Denial of Service Attack Since No Restriction in Access.

There is no Confirmation of Seat Allocation in the Project.

There is Fees Information Provided in the Project.

It is Prediction of Possibility of Admission But Specific University Prediction is not Provided.

Students might get Demotivated on seeing Their Negative Results.

# 11 CONCLUSION

# 12 FUTURE SCOPE

In Future we will be able to Train the Machine Learning Model with Institution Specific Data , So That we will be able to check if the Student is able to Join the Institution based on all Cirucumstances such as Distance From Home , Fees Collected by the Institution etc. and so The Suggestions we be Provided on Which College is most Probable in case The College they Requested has No Chances.

# 13 APPENDIX

## Source Code

```
from flask import Flask, render_template, request
from flask_cors import CORS
import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "PLACE THE API KEY HERE"

token_response = requests.post("https://iam.cloud.ibm.com/identity/token",data = {
    "apikey":API_KEY , "grant_type":"urn:ibm:params:oauth:grant-type:apikey"
})

mltoken = token_response.json()["access_token"]

header = {"Content-Type" : "application/json" , "Authorization" : "Bearer " + mltoken}



app = Flask(__name__,static_url_path='')
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
```

```
# No cacheing at all for API endpoints.
@app.after_request
def add_header(response):
    # response.cache_control.no_store = True
    if 'Cache-Control' not in response.headers:
        response.headers['Cache-Control'] = 'no-store'
    return response

CORS(app)

@app.route('/',methods=['GET','POST'])
def sendHomePage():
    if(request.method == "GET"):
        return render_template('index.html' , notification = 'Welcome!')
    else:
        return render_template('index.html' , notification = 'You have given ' +
request.form['rating'] + ' Stars Feedback')

@app.route('/predict',methods=['POST'])
def PredictPossibility():
    GREScore = float(request.form['GREScore'])
    TOEFLScore = float(request.form['TOEFLScore'])
    UnivRating = float(request.form['UnivRating'])
    SOP = float(request.form['SOP'])
    LOR = float(request.form['LOR'])
    CGPA = float(request.form['CGPA'])
    Research = 0
    if('Research' in request.form):
        Research = 1
    X = [[GREScore , TOEFLScore , UnivRating , SOP , LOR , CGPA , Research ]]
    print(X)
    # NOTE: manually define and pass the array(s) of values to be scored in the next line
    payload_scoring = {"input_data": [{"fields":
[["GREScore","TOEFLScore","UnivRating","SOP","LOR","CGPA","Research"]], "values":
X}]}
    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/1b254abd-eaec-4f59-9e86-
```

```
a3674f3eea4c/predictions?version=2022-11-15', json=payload_scoring,
    headers=header)
    print("Scoring response")
    print(response_scoring.json())
    probability = int(round((response_scoring.json()["predictions"][0]["values"][0][0]),2)*100)
    print(probability)
    prob_comment = ""
    color_scheme = ""
    if(probability > 100):
        probability = 100
    elif(probability < 0):
        probability = 0
    if(probability < 50):
        prob_comment = "The Chances of Getting an Admission is less likely"
        color_scheme = 'darkorange'
    elif(probability < 70):
        prob_comment = "There is a slight Chance of Possibility."
        color_scheme = 'yellow'
    else:
        prob_comment = "There is High Chances of Possibility"
        color_scheme = 'lawngreen'
    return
render_template('predict.html',predict=probability,comment=prob_comment,color_scheme=co
lor_scheme)

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

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" type="text/css" href="{{ url_for( 'static' , filename='css/styles.css'
)}}" />
    <title>University Admit Eligibility Predictor</title>
```

```
</head>
<body>
   <div id="notification" >
      <p id="message">{{notification}}</p>
   </div>
   <h1 id="projName">University Admit Eligibility Predictor</h1>
   <form id="eligibilityForm" method="POST" action="/predict">
      <div class="inputBox">
         <p class="inputLabel">GRE Score</p>
         <input class="inputField" type="number" name="GREScore" min="260" max="340"
placeholder="260-340" required/>
      </div>
      <div class="inputBox">
         <p class="inputLabel">TOEFL Score</p>
         <input class="inputField" type="number" name="TOEFLScore" min="0" max="120"
placeholder="0-120" required/>
      </div>
      <div class="inputBox">
         <p class="inputLabel">University Rating</p>
         <input class="inputField" type="number" name="UnivRating" min="0" max="5"
placeholder="0-5" required/>
      </div>
      <div class="inputBox">
         <p class="inputLabel">Statement of Purpose</p>
         <input class="inputField" type="number" name="SOP" min="0" max="5" step="0.5"
placeholder="0-5" required/>
      </div>
      <div class="inputBox">
         <p class="inputLabel">Letter of Recommendation</p>
         <input class="inputField" type="number" name="LOR" min="0" max="5" step="0.5"
placeholder="0-5" required/>
      </div>
      <div class="inputBox">
         <p class="inputLabel">CGPA</p>
         <input class="inputField" type="number" name="CGPA" min="5" max="10"
step="0.01" placeholder="0-10" required/>
      </div>
```

```
        <div class="inputBox">
            <p class="inputLabel">Completed a Research</p>
            <input class="inputField" type="checkbox" name="Research" value='done'/>
        </div>
        <div class="inputBox">
            <div class="emptyDiv"></div>
            <button id="submitButton" class="inputField" type="submit">Submit</button>
        </div>
    </form>
    <script src="{{ url_for('static',filename='js/index.js' )}}" ></script>
</body>
</html>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/styles.css') }}"
/>
    <script src="{{ url_for('static', filename='js/index.js') }}" type="text/javascript" ></script>
    <link rel="stylesheet"
href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined:opsz,wght,FILL
,GRAD@20..48,100..700,1,-50..200" />
    <title>Results</title>
</head>
<body>
    <h1 id="projName">University Admit Eligibility Predictor</h1>
    <div id="pageSection">
        <h1 id="pageTitle">Results</h1>
        <h1 id="resultText">The Possibility of Admission is : </h1>
        <h1 id="probability" style="color:{{color_scheme}}">{{predict}}%</h1>
        <h1 id="comment" style="color:{{color_scheme}}">{{comment}}</h1>
        <form id="feedbackForm" action="{{url_for('sendHomePage')}}" method="POST" >
            <fieldset id="feedback" >
                <div id="feedbackEmoji">
```

```
                <label for="fiveStar">
                    <input type="radio" name="rating" id="fiveStar" value="5" checked >
                    <span class="material-symbols-outlined">
                        sentiment_very_satisfied
                    </span>
                </label>
                <label for="fourStar">
                    <input type="radio" name="rating" id="fourStar" value="4" >
                    <span class="material-symbols-outlined">
                        mood
                    </span>
                </label>
                <label for="threeStar">
                    <input type="radio" name="rating" id="threeStar" value="3" >
                    <span class="material-symbols-outlined">
                        sentiment_satisfied
                    </span>
                </label>
                <label for="twoStar">
                    <input type="radio" name="rating" id="twoStar" value="2" >
                    <span class="material-symbols-outlined">
                        sentiment_dissatisfied
                    </span>
                </label>
                <label for="oneStar">
                    <input type="radio" name="rating" id="oneStar" value="1" >
                    <span class="material-symbols-outlined">
                        sentiment_extremely_dissatisfied
                    </span>
                </label>
            </div>
            <input type="submit" id="feedbackSubmit" class="inputField" name="Submit"
value="Submit" />
        </fieldset>
    </form>
    <form id="returnForm" action="{{url_for('sendHomePage')}}" method="GET" >
        <button id="submitButton" class="inputField">Go Back</button>
```

```
        </form>
    </form>
</body>
</html>

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
get_ipython().run_line_magic('matplotlib', 'inline')
df = pd.read_csv('dataset/Admission_Predict.csv')
df.head()
df.drop(["Serial No."],axis = 1,inplace = True)
df.head()
df.describe()
df.info()
df.isnull().sum()
sns.distplot(df['TOEFL Score'])
plt.pie(df['University Rating'].value_counts(),[0.05,0.05,0.05,0.05,0.05],df['University
Rating'].value_counts().index,autopct='%1.1f%%')
plt.title('University Rating')
plt.plot()
sns.barplot(df.Research.value_counts().index,df.Research.value_counts())
sns.lineplot(df['GRE Score'],df['Chance of Admit'])
def countplot_2(x,hue,title=None,figsize=(12,10)):
    plt.figure(figsize=figsize)
    sns.countplot(data = df[[x,hue]],x=x,hue=hue)
    plt.title(title)
    plt.plot()
countplot_2('Research','University Rating','University based Research Ability')
sns.pairplot(df)
df.corr()
plt.figure(figsize=(12,10))
sns.heatmap(df.corr(),annot=True)
plt.show()
for i in df.columns:
    sns.boxplot(df[i])
```

```
    plt.show()
for i in df.columns:
    Q1 = df[i].quantile(0.25)
    Q3 = df[i].quantile(0.75)
    IQR = Q3-Q1
    upper_limit = Q3 + (1.5*IQR)
    lower_limit = Q1 - (1.5*IQR)
    df[i] = np.where(df[i]>=upper_limit,Q3 + (1.5*IQR),df[i])
    df[i] = np.where(df[i]<=lower_limit,Q1 - (1.5*IQR),df[i])
for i in  df.columns:
    sns.boxplot(df[i])
    plt.show()
X = df.drop(columns=['Chance of Admit'])
X
Y = pd.DataFrame(df['Chance of Admit'])
Y
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
X_Scaled = pd.DataFrame(scale.fit_transform(X),columns = X.columns)
target_scale = MinMaxScaler()
Y_Scaled = pd.DataFrame(target_scale.fit_transform(Y),columns = Y.columns)
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size = 0.2,random_state=90)
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train,y_train)
y_test.to_numpy().flatten()
y_predict = model.predict(x_test)
pd.DataFrame({'Actual':y_test.to_numpy().flatten(),'Predicted':y_predict.flatten()})
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
print(r2_score(y_test,y_predict))
print(mean_absolute_error(y_test,y_predict))
print(mean_squared_error(y_test,y_predict))
print(mean_squared_error(y_test,y_predict,squared=False))
import joblib
```

```
joblib.dump(model,r'model/Admission_Predictor.pkl')
```

# GitHub & Project Demo Link

Github Link - https://github.com/IBM-EPBL/IBM-Project-19561-1659700246

Project Demo Link - https://drive.google.com/drive/folders/1P8yqJCmuqwywj_XoR6rtEOR-iseXqTDn?usp=sharing