UNIVERSITY ADMIT ELIGIBILITY PREDICTOR PROJECT REPORT

TEAM ID: PNT2022TMID53103

Submitted by

Sahana N

Prasanna Hari

Pydipati Nikhitha

Shruti Susila Prasanna

SSN College of Engineering

NOV 2022

Contents

1. Introduction

Project Overview

University Admit Eligibility Predictor Students are often worried about their chances of admission to University. The aim of this project is to help students in shortlisting universities with their profiles. The predicted output gives them a fair idea about their admission chances in a particular university of a particular rank. This analysis should also help students who are currently preparing or will be preparing to get a better idea

Purpose

The University Admission process can be quite a long and tedious process for anyone. There’s multiple criteria and it is ever changing for multiple universities of different ranks. This web based application will take the input of multiple criterias to predict the chances of admission for the respective student.

2.Literature Survey

Existing Problem

University admissions criteria are so vast and different for many universities.

How can we predict whether a student will get an admit or not? What are the parameters for selection? Can it be mathematically expressed? This is the main existing problem

References

1. “A Machine Learning Approach for Graduate Admission Prediction”
AUTHORS : Amal AlGhamdi, Amal Barsheed, Hanadi AlMshjary,
Hanan AlGhamdi

2. “Predicting Student University Admission Using Logistic
Regression” AUTHORS : Sharan Kumar Paratala Rajagopal

3. Applying A Hybrid Model Of Neural Network And Decision Tree Classifier For Predicting University Admission

AUTHORS : Simon Fong,Yain-Whar Si,Robert P.Biuk-Aghai

4. Design and Implementation of a Hybrid Recommender System for Predicting College Admission

AUTHORS : Abdul Hamid M. Ragab, Abdul Fatah S. Mashat, Ahmed M. Khedra

5. A Comparative Study on University Admission Predictions Using Machine Learning Techniques

AUTHORS : Prince Golden, Kasturi Mojesh, Lakshmi Madhavi Devarapalli, Pabbidi Naga Suba Reddy, Srigiri Rajesh, Ankita Chawla

6. Deep Learning in diverse Computing and Network Applications

Student Admission Predictor using Deep Learning AUTHORS : P. Nandal

Problem Statement Definition

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A student	Get into the university of my choice	I am not sure whether I will be admitted	I lack knowledge regarding the admission process and criteria	Anxious
PS-2	A teacher	Help my students get into the university of their choice	I am not sure whether I am doing my best to ensure their admission	I need more knowledge regarding the admission process and criteria to guide them	Concerned
PS-3	a Career consultancy service	Guide multiple clients to get admitted into the university of their choice	I cannot keep up with the changing university requirements for multiple universities	The university eligibility criteria keeps changing for many colleges every year	Overwhelmed

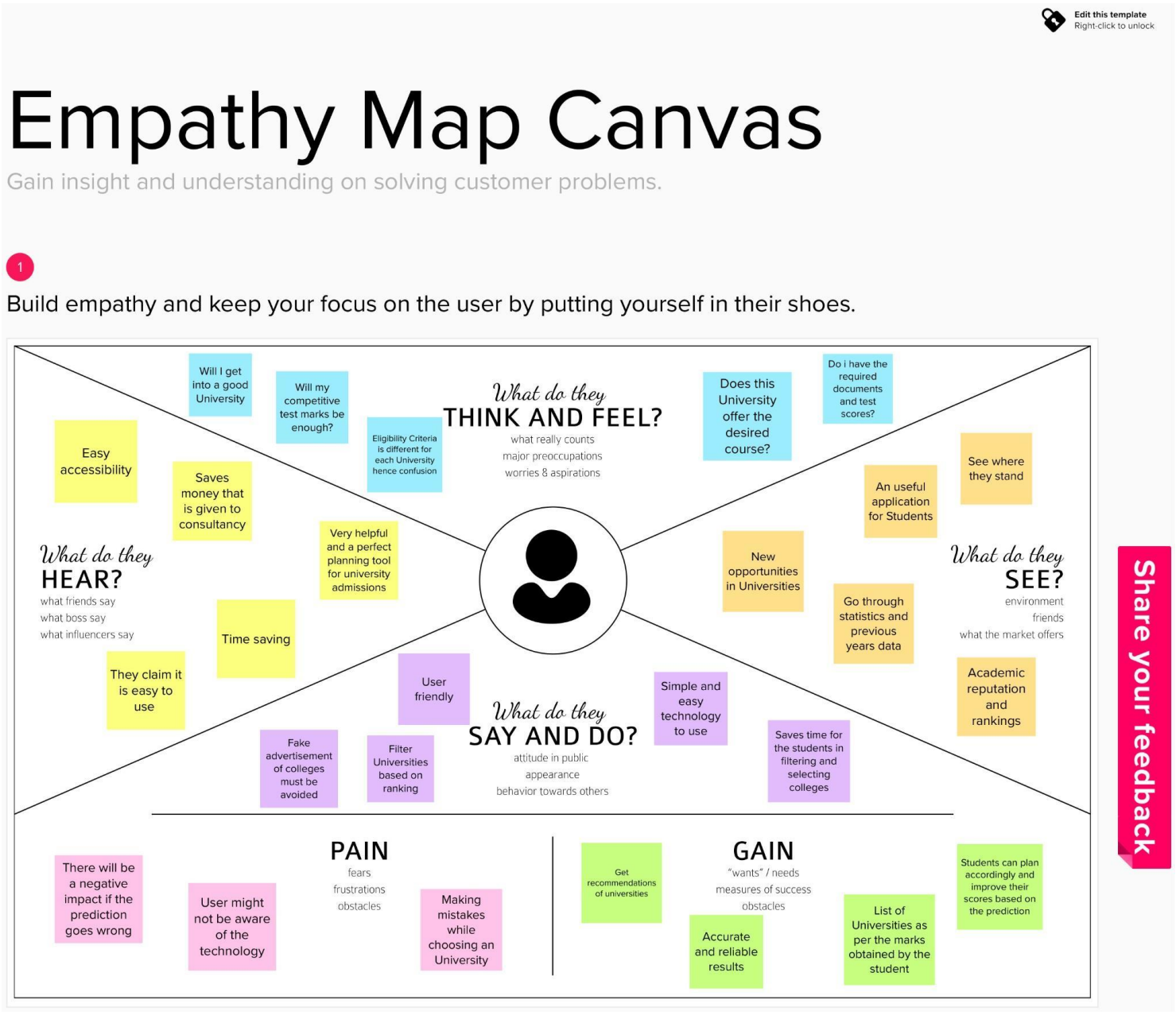
Ideation Phase
Empathize & Discover

Date	19 September 2022
Team ID	PNT2022TMID53103
Project Name	Project - University Admit Eligibility Predictor
Maximum Marks	4 Marks

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to help teams better understand their users.
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



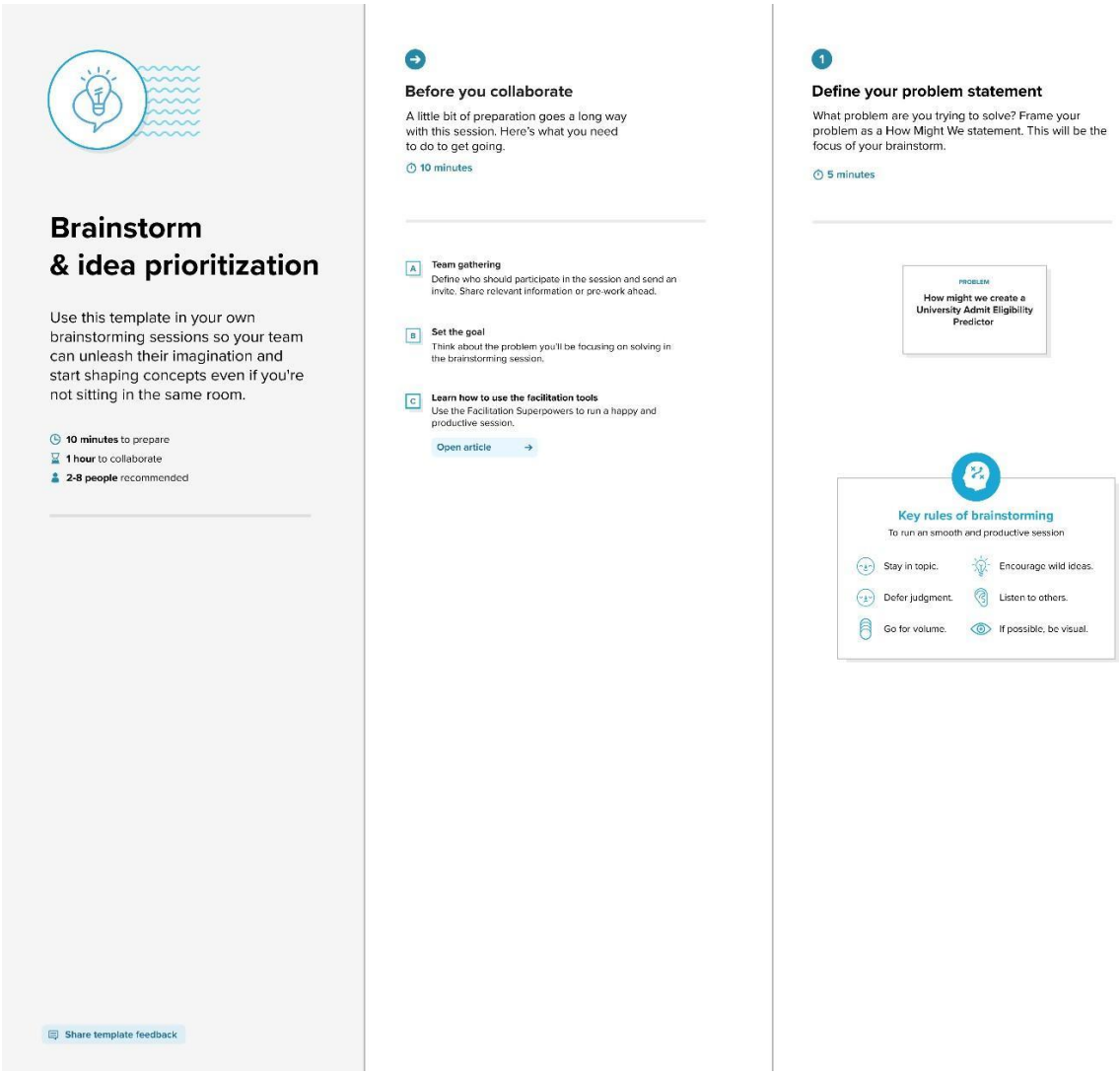
Ideation Phase
Brainstorm & Idea Prioritization Template

Date	19 September 2022
Team ID	PNT2022TMID53103
Project Name	Project – University Admit Eligibility Predictor
Maximum Marks	4 Marks

Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Shruti

- Web-based application using python flask,html,etc
- The proposed solution should have good time complexity
- User can upload their own real-time dataset for analyzing as a csv or excel file
- It examines user requirements and works in that direction

Sahana

- Preprocess the data and remove dirty data and replace missing values
- Equiped with latest ML and data science technologies
- Past college admission criteria are used to predict future admissions
- Chance of Admit of University Admission is displayed

Prasanna

- Criteria of different university are checked at regular intervals
- Crucial to maintain privacy and security in application
- Evaluation metrics of model such as accuracy,F1 score etc should be satisified
- Feedback for improvement of Admission profile is given

Nikhitha

- Different test scores,grades and essays inputs are taken into consideration
- A logistic regression model will be applied for analysis
- User can get one time prediction without storing the data in the application
- User friendly interface

3

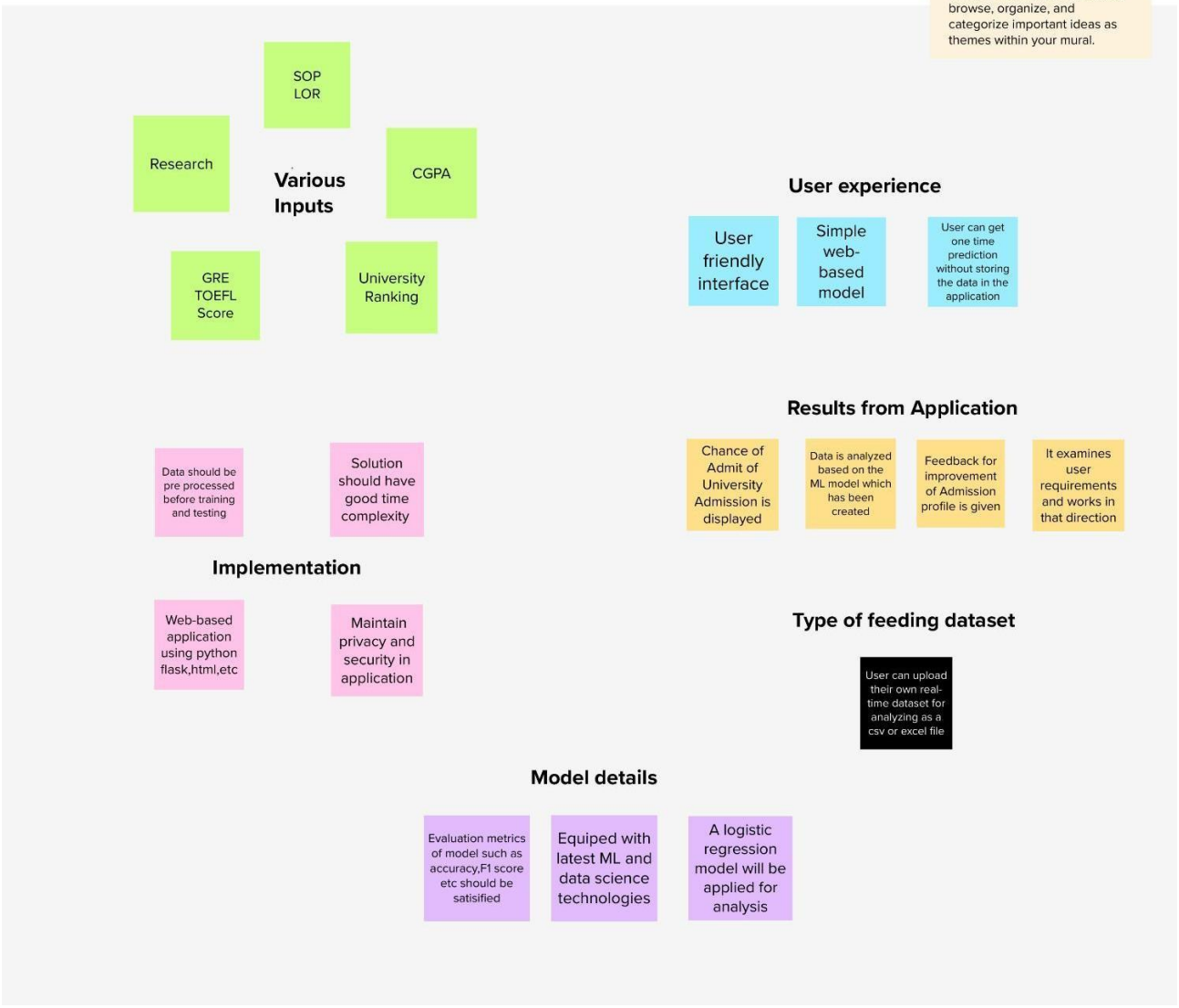
Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.



Step-3: Idea Prioritization

4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

PROJECT DESIGN PHASE-I

PROPOSED SOLUTION TEMPLATE

Date	15 October 2022
Team ID	PNT2022TMID53103
Project Name	Project - University Admit Eligibility Predictor
Maximum Marks	2 Marks

PROPOSED SOLUTION TEMPLATE :

Project team shall fill the following information in the proposed solution template.

S.NO.	PARAMETER	DESCRIPTION
1.	Problem Statement (Problem to be solved)	University admissions is a very competitive field that is ever-changing. This causes students to worry a lot about their admissions.The aim of this project is to help students in shortlisting universities with their profiles. The predicted output gives them a fair idea about their admission chances in a particular university. The project helps students in search of colleges as well as those currently preparing.
2.	Idea / Solution description	We collect several data points such as CGPA,SOP,LOR,GRE,TOEFL score etc.,. in order to examine the chances a student has to enter a given University. An updated database of different universities and their admit criteria is maintained in order to cross check and confirm the University Admit Eligibility.
3.	Novelty / Uniqueness	University students find it difficult to get easy access to predictors and the accuracy of the predictions are often compromised. This model creates an accurate prediction regarding the eligibility of a given student in a university.

4.	Social Impact / Customer Satisfaction	Better interface for student satisfaction. Accurate results at good time complexity Large database to ensure checks for different universities.

		Better interaction between student and application
5.	Business Model (Revenue Model)	The model serves as a revenue source to career consultancy services and schools aiming to have a large university turnover rate. The easy and interactive UI makes it a selling point for all interested parties.
6.	Scalability of the Solution	The model can be expanded easily to include other data points based selectively on different regions and Universities. Region specific data can be added to the database for eligibility checks to ensure worldwide usage of the solution.

Project Title: University Admit Eligibility Predictor
PNT2022TMID53103

Project Design Phase-I - Solution Fit Template

Team ID:

Identify strong TR & EM

3. TRIGGERS

TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

1) Concern about future
2) Peer pressure
3) Critical time period

4. EMOTIONS: BEFORE / AFTER

EM

How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

Before: Anxious, Stressed, Concerned
After: Relief, Confidence, Satisfaction

10. YOUR SOLUTION

SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

A user -friendly application which uses latest ML models to predict the chance of admission with high accuracy and good time complexity

8. CHANNELS of BEHAVIOUR

CH

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

Upload data into the predictor

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

Gathers data on their academic profile

Project Design Phase-II
Solution Requirements (Functional & Non-functional)

Date	15 October 2022
Team ID	PNT2022TMID53103
Project Name	Project - University Admit Eligibility Predictor
Maximum Marks	4 Marks

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	**User Registration**	● Registration through Form ● Registration through Gmail ● Registration through LinkedIN
FR-2	**User Confirmation**	● Confirmation via Email ● Confirmation via OTP
FR-3	**User Details**	● Submit the documents ● GRE or/and TOEFL score sheet ● Curriculum Vitae (CV) ● Statement of Purpose (SoP) ● Letter of Recommendation
FR-4	**User Requirements**	● Upload all the relevant documents in the appropriate location in the website ● The list of all possible university for the candidate would be displayed based on the information

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	● The system doesn't expect any technical pre-requisite from the user i.e.; even the naive user can access it ● User friendly ● The UI would focus on recognize over recall
NFR-2	**Security**	● User with valid login credentials will be able to access the site

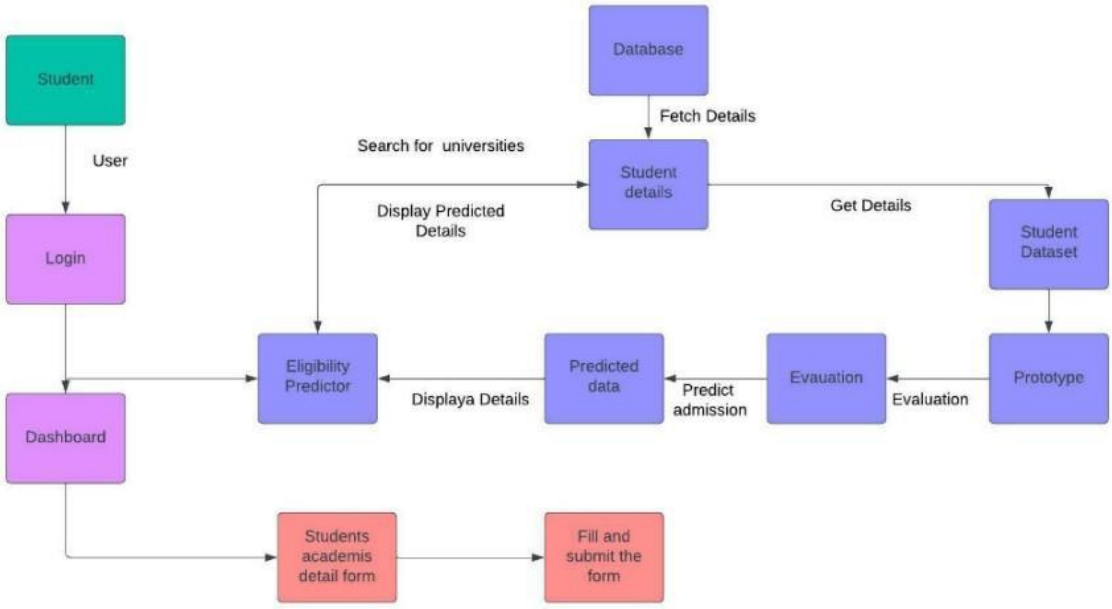
		• Under any error, the system should be able to come back to normal operation in under an hour
NFR-3	**Reliability**	• Model accuracy is high • The system would always strive for maximum reliability due to the importance of data and damages that could be caused by incomplete and incorrect data
NFR-4	**Performance**	• The website can efficiently handle the traffic by servicing the request as soon as possible • The response time is low
NFR-5	**Availability**	• Minimal data redundancy • Fast and efficient • The system will run 7 days a week, 24 hours a day
NFR-6	**Scalability**	• Works well under multiple requests

Project Design Phase-II
Data Flow Diagram & User Stories

Date	03 October 2022
Team ID	PNT2022TMID53103
Project Name	Project - University Admit Eligibility Predictor
Maximum Marks	4 Marks

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Student (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive a confirmation email once I have registered for the application.	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook.	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail.	I can register & access the dashboard with Gmail login.	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password.	I can log into my account	High	Sprint-1
	Dashboard	USN-6	On entering the page, I can see the homepage, logout option and student details.	I can view all components and select what is necessary.	High	Sprint-1
Student (Web user)	Registration	USN-7	As a user, I can register on the website by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
	Login	USN-8	As a user, I can log into the website by entering email & password.	I can log into my account	High	Sprint-1
Administrator	Overview	USN-9	As an admin, I can view all the searches for eligibility.	I can view all searches done so far.	Medium	Sprint-2
	Editing	USN-10	As an admin, I can edit working and details of the universities.	I can edit any component and working of the model.	High	Sprint-1

Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	18 October 2022
Team ID	PNT2022TMID53103
Project Name	Project - University Admit Eligibility Predictor
Maximum Marks	4 Marks

Technical Architecture:

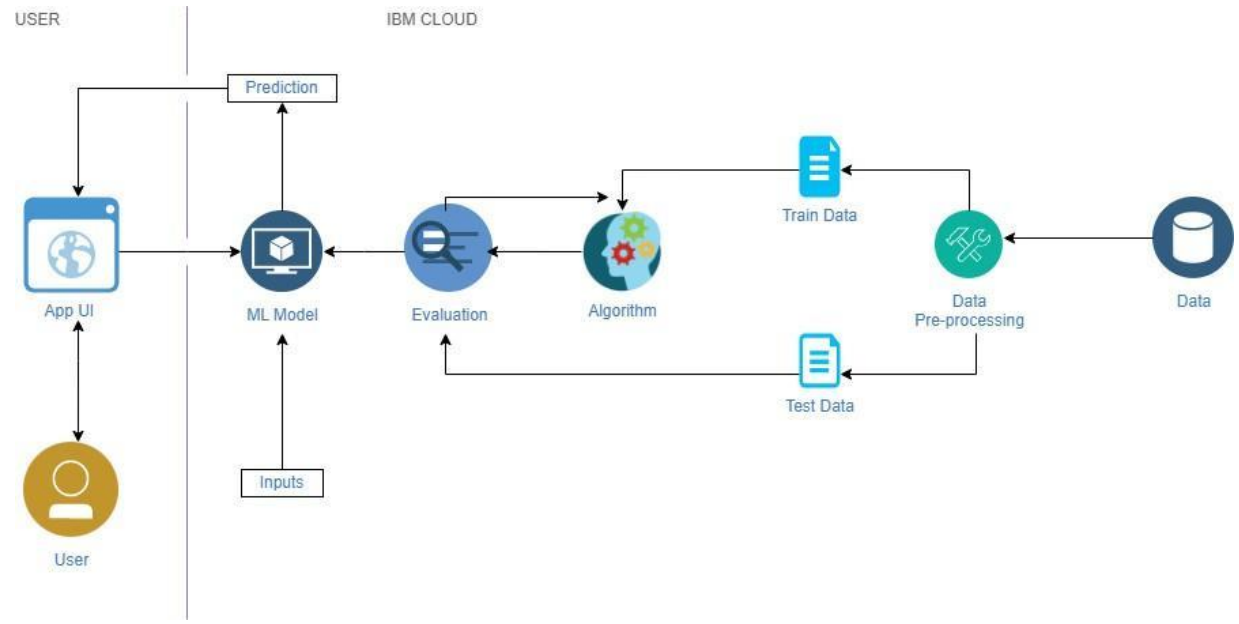


Table – 1: Components & Technologies

S.No	Component	Description	Technology
1.	User Interface	User interacts with the application by using Web UI	HTML, CSS, JS
2.	Application Logic - 1	Collecting input from user	Python

3.	Application Logic – 2	Integrating front-end and back-end	Flask
4.	Application Logic – 3	Training and testing of model	IBM Watson
5.	Cloud Database	Database service on cloud	IBM DB2
6.	Machine Learning Model	Predicting chance of admission into university	Logistic Regression

Table - 2: Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask is used in back-end	Python
2.	Security Implications	Http authentication	Flask Security
3.	Scalable Architecture	Works well under multiple requests	IBM Watson
4.	Availability	Available all the time	IBM Watson
5.	Performance	Time required to predict chance of admission	Machine Learning

Project Design Phase-I
Solution Architecture

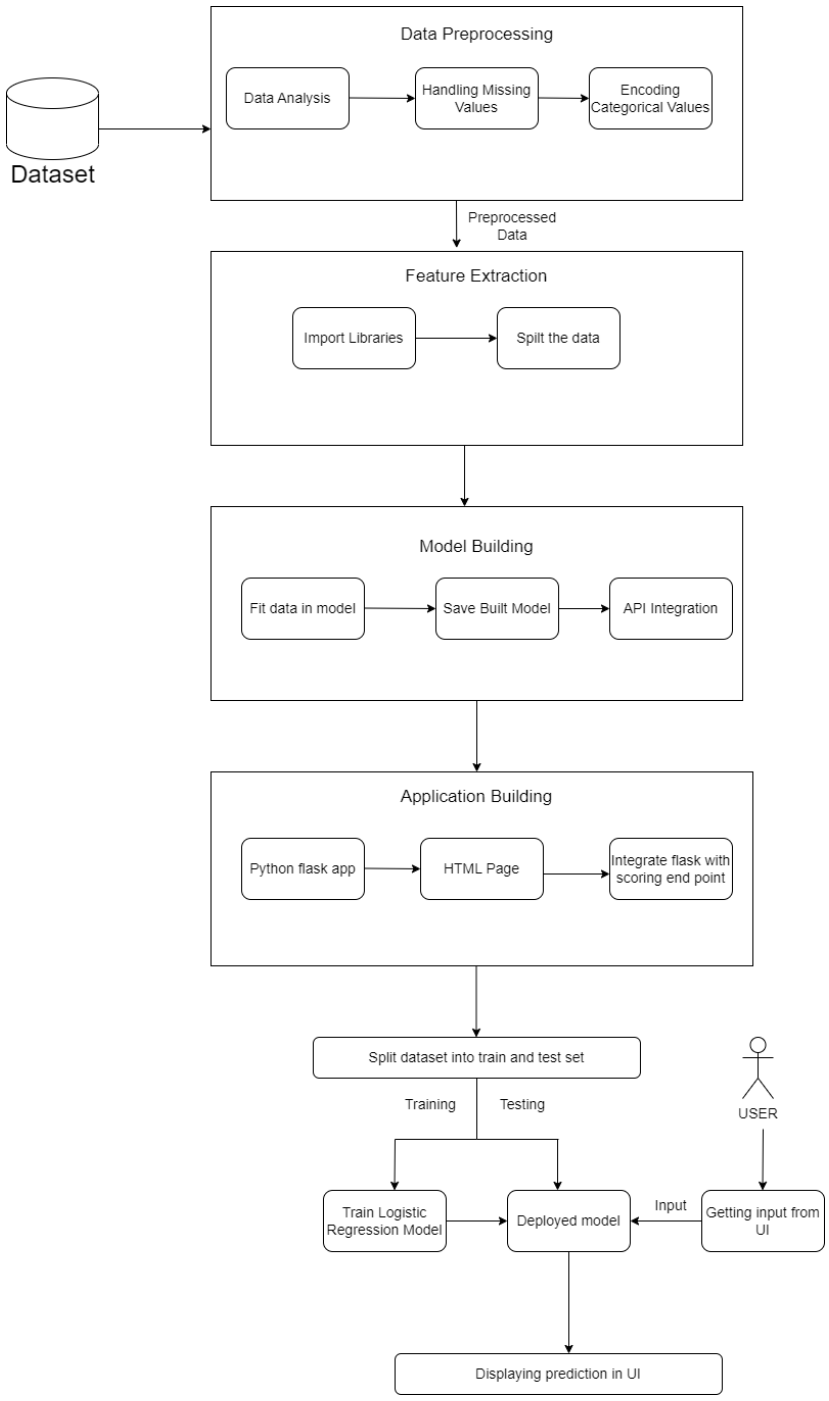
Date	17 October 2022
Team ID	PNT2022TMID53103
Project Name	Project -University Admit Eligibility Predictor
Maximum Marks	4 Marks

User View:

1. User enters the academic details in the UI
2. Entered input is sent to the regression model deployed through IBM Watson
3. The model predicts the chance of admission and sends it to the UI
4. The predicted value is displayed at the frontend

Model View:

1. The dataset is pre-processed for handling missing values/categorical values
2. Feature extraction is performed
3. The data is split into dependent and independent variables
4. The dataset is split as train and test set
5. A logistic regression model is built and is trained with the training data
6. The model is evaluated using the testing data
7. The trained model is deployed in IBM Watson


Data Preprocessing
Data Analysis
Handling Missing Values
Encoding Categorical Values
Dataset
Preprocessed Data
Feature Extraction
Import Libraries
Spilt the data
Model Building
Fit data in model
Save Built Model
API Integration
Application Building
Python flask app
HTML Page
Integrate flask with scoring end point
Split dataset into train and test set
Training
Testing
USER
Train Logistic Regression Model
Deployed model
Input
Getting input from UI
Displaying prediction in UI

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	18 October 2022
Team ID	PNT2022TMID53103
Project Name	Project - University Admit Eligibility Predictor
Maximum Marks	8 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user , I need to collect the data and design User Interface	2	High	Sahana Nikhitha
Sprint-1		USN-2	As a user , I need to collect the data and design User Interface	1	High	Prasanna Shruti
Sprint-1		USN-3	As a user , I need to collect the data and design User Interface	2	Low	Shruti Nikhitha
Sprint-1		USN-4	As a user , I need to collect the data and design User Interface	3	Medium	Sahana Prasanna
Sprint-2	User Action	USN-5	As a user, I can access dataset of multiple universities	1	High	Sahana Nikhitha Shruti
Sprint-2	Model Building	USN-6	As a user I need the machine learning model to pre-process the data	3	High	Nikhitha Prasanna
Sprint-3	Model Training and Testing	USN-7	As a user ,I need the machine learning model to predict the chance of admission	3	High	Sahana Shruti
Sprint-3	Model Training and Testing	USN-8	As a user ,I need the machine learning model to predict the chance of admission	2	Medium	Prasanna Sahana Nikhitha Shruti
Sprint-4	Deployment	USN-9	As a user , I need the application to be accessible all over the world	1	Medium	Shruti Prasanna

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-4	Deployment	USN-10	As a user , I need the application to be accessible all over the world	2	Medium	Nikhitha Shruti

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	7 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	10 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	13 Nov 2022

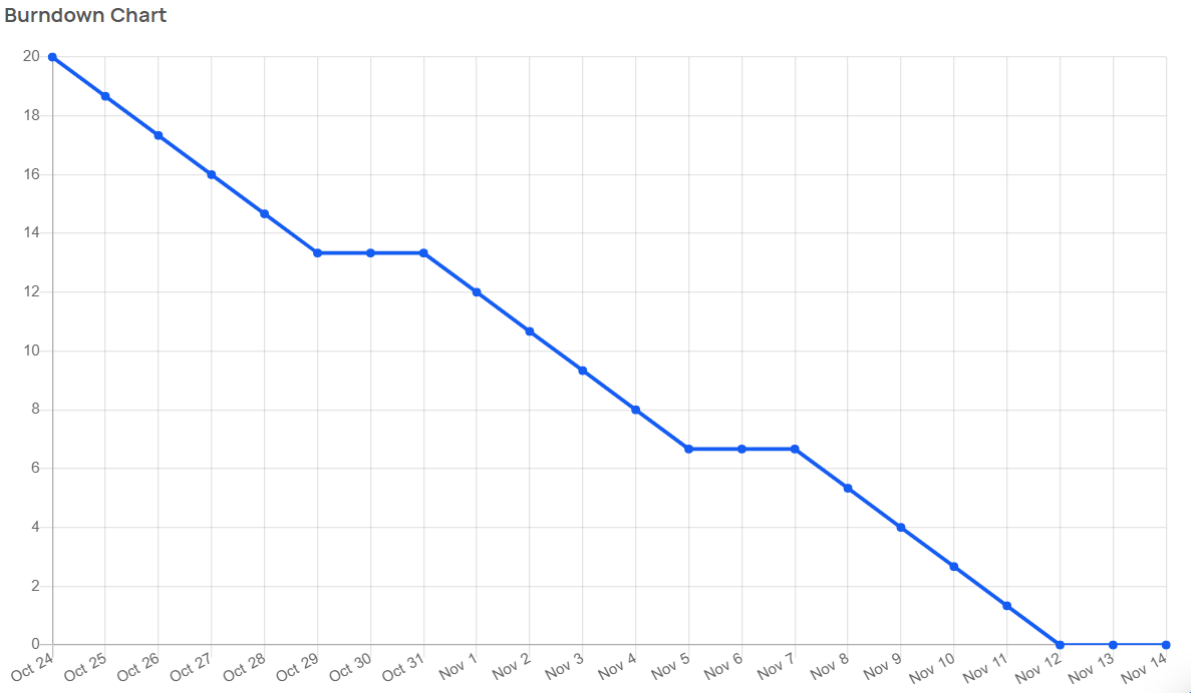
Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

Burndown Chart:

A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



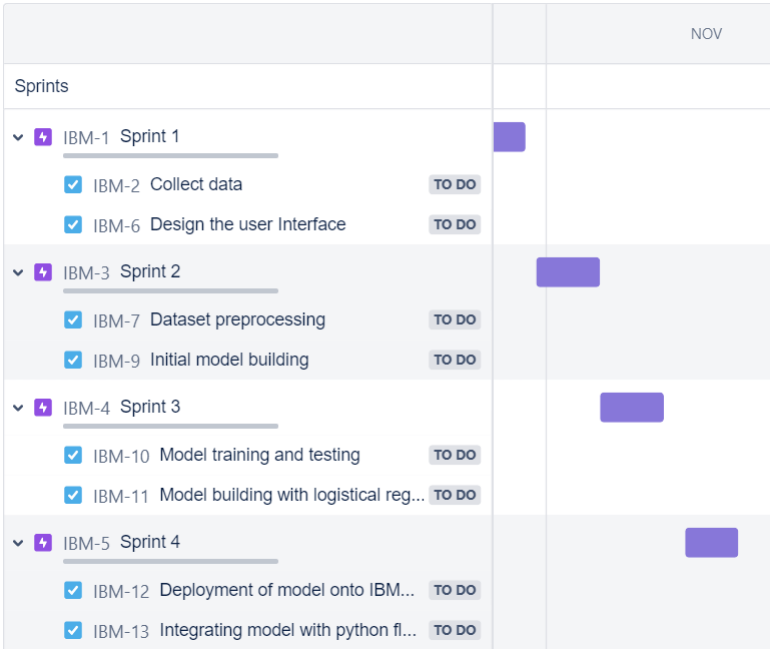
https://www.visual-paradigm.com/scrum/scrum-burndown-chart/
https://www.atlassian.com/agile/tutorials/burndown-charts

Reference:

https://www.atlassian.com/agile/project-management
https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software https://www.atlassian.com/agile/tutorials/epics
https://www.atlassian.com/agile/tutorials/sprints
https://www.atlassian.com/agile/project-management/estimation
https://www.atlassian.com/agile/tutorials/burndown-charts

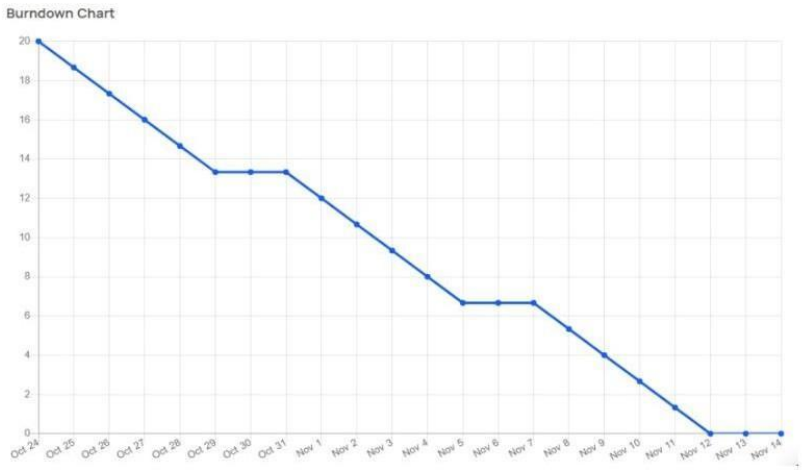
Jira Files

Road Map

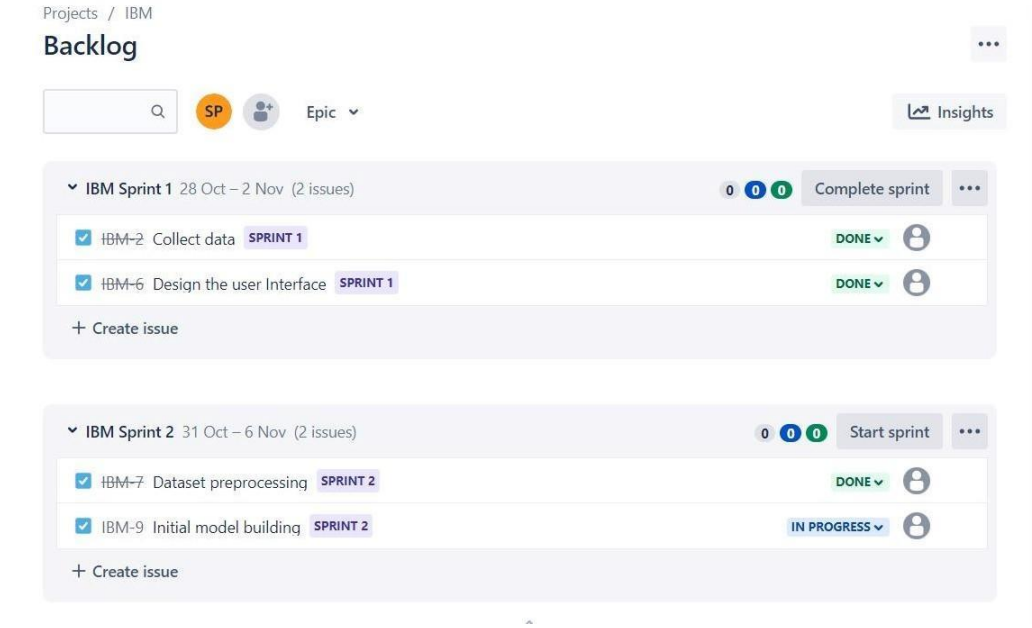


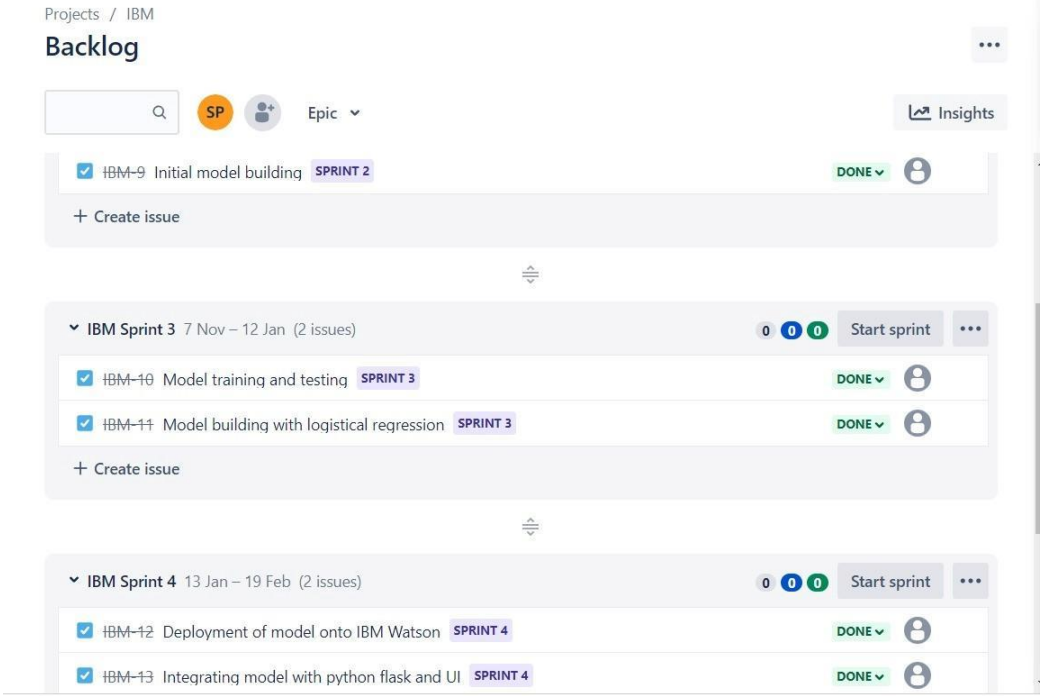
Burndown Chart:

A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



BackLogs


Projects / IBM
Backlog
SP
Epic
Insights
IBM Sprint 1 28 Oct – 2 Nov (2 issues)
0 0 0
Complete sprint
IBM-2 Collect data SPRINT 1
DONE
IBM-6 Design the user Interface SPRINT 1
DONE
+ Create issue
IBM Sprint 2 31 Oct – 6 Nov (2 issues)
0 0 0
Start sprint
IBM-7 Dataset preprocessing SPRINT 2
DONE
IBM-9 Initial model building SPRINT 2
IN PROGRESS
+ Create issue


Projects / IBM
Backlog
SP
Epic
Insights
IBM-9 Initial model building SPRINT 2
DONE
+ Create issue
IBM Sprint 3 7 Nov – 12 Jan (2 issues)
0 0 0
Start sprint
IBM-10 Model training and testing SPRINT 3
DONE
IBM-11 Model building with logistical regression SPRINT 3
DONE
+ Create issue
IBM Sprint 4 13 Jan – 19 Feb (2 issues)
0 0 0
Start sprint
IBM-12 Deployment of model onto IBM Watson SPRINT 4
DONE
IBM-13 Integrating model with python flask and UI SPRINT 4
DONE

7. Coding & Solutioning

7.1 Feature 1

This feature will predict the chances of admission in the university. The feature was designed in the html code connected with app.py as the backend.

chance.html

```
<!DOCTYPE html>
<html>
<head>
<title>University Admit Eligibility Predictor</title>
<link rel = "icon" href = "/static/img/Logo.jpg" type = "image/x-icon">
<body style=" background-image:url('/static/img/background 1.jpg');background-repeat:
no-repeat;background-attachment:fixed;background-size: 100% 100%;" >
<div style="position:absolute;left:1300px">
<img src="/static/img/Logo.jpg" alt="Website Logo" style="width:40px;height:40px;">
</div>
<div style="position:absolute;top:20px;left:280px">
<p style="font-family:serif;font-size:45px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<strong>University Admit Eligibility Predictor</strong>
<div style="position:absolute;top:70px;left:200px">
<p style="font-family:serif;font-size:80px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<p style="color: black"><i><strong>You have a good chance of admit!
Congratulations!</strong><br>{{content}}%</i></p>

</p>
</div>
</body>
</html>
```

7.2 Feature 2

This feature will predict the low chances of admission in the university. The feature was designed in the html code connected with app.py as the backend.

nochance.html

```
<!DOCTYPE html>
<html>
<head>
<title>University Admit Eligibility Predictor</title>
<link rel = "icon" href = "/static/img/Logo.jpg" type = "image/x-icon">
<body style=" background-image:url('/static/img/background 1.jpg');background-repeat:
no-repeat;background-attachment:fixed;background-size: 100% 100%;" >
<div style="position:absolute;left:1300px">
```

```
<img src="/static/img/Logo.jpg" alt="Website Logo" style="width:40px;height:40px;">
</div>
<div style="position:absolute;top:20px;left:280px">
<p style="font-family:serif;font-size:45px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<strong>University Admit Eligibility Predictor</strong>
<div style="position:absolute;top:70px;left:200px">
<p style="font-family:serif;font-size:80px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<p style="color: black"><i><strong>You do not have a good chance of admit!
Sorry!</strong><br>{{content}}%</i></p>

</p>
</div>
</body>
</html>
```

app.py

```
from flask import Flask, render_template, redirect, url_for, request,jsonify
import requests

API_KEY = "_jEZ9faTKtRDlZDFHcrrAbuQFdvxbBeiu8W1D8-4z7ng"
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__)

@app.route("/")
def home():
    return render_template("Main Page.html")


@app.route("/Page 1",methods = ['POST','GET'])
def input():
    return render_template("Page 1.html")

@app.route("/", methods = ['POST','GET'])
def prediction():
    if request.method == 'POST':
        arr = []
        for i in ["GRE Score","TOEFL Score","University
Rating","SOP","LOR","CGPA","Research"]:
            val = request.form[i]
            if val == '':
                return redirect(url_for("Page 1.html"))
            arr.append(float(val))
        #print(arr)
        # deepcode ignore HardcodedNonCryptoSecret: <please specify a reason of
ignoring this>

        payload_scoring = {
```

```
            "input_data": [{"fields":[  'GRE Score',
                                        'TOEFL Score',
                                        'University Rating',
                                        'SOP',
                                        'LOR ',
                                        'CGPA',
                                        'Research'],
                            "values": [arr]
                            }]
                        }

        response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/ed4ff532-801c-457b-85b9-
ffa103d3a064/predictions?version=2022-11-18',
json=payload_scoring,headers=header).json()
        print(response_scoring)
        result = response_scoring['predictions'][0]['values']
        print(result)

        if result[0][0] > 0.5:
            return redirect(url_for('chance', percent=result[0][0]*100))
        else:
            return redirect(url_for('no_chance', percent=result[0][0]*100))
    else:
        return redirect(url_for("Page 1"))
    return ""




@app.route("/chance/<percent>")
def chance(percent):
    return render_template("chance.html", content=[percent])

@app.route("/nochance/<percent>")
def no_chance(percent):
    return render_template("noChance.html", content=[percent])



@app.route('/<path:path>')
def catch_all(path):
    return render_template("Page 1.html")

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

7.3 Database Schema

The database used in this project was Admission_Predict.csv. Sample screenshot of the database:

401 lines (401 sloc) | 12.6 KB

Raw Blame

Search this file...

Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
1	337	118	4	4.5	4.5	9.65	1	0.92
2	324	107	4	4	4.5	8.87	1	0.76
3	316	104	3	3	3.5	8	1	0.72
4	322	110	3	3.5	2.5	8.67	1	0.8
5	314	103	2	2	3	8.21	0	0.65
6	330	115	5	4.5	3	9.34	1	0.9
7	321	109	3	3	4	8.2	1	0.75
8	308	101	2	3	4	7.9	0	0.68
9	302	102	1	2	1.5	8	0	0.5
10	323	108	3	3.5	3	8.6	0	0.45
11	325	106	3	3.5	4	8.4	1	0.52
12	327	111	4	4	4.5	9	1	0.84
13	328	112	4	4	4.5	9.1	1	0.78
14	307	109	3	4	3	8	1	0.62
15	311	104	3	3.5	2	8.2	1	0.61
16	314	105	3	3.5	2.5	8.3	0	0.54

8. Testing

8.1 Test Cases

report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	50	0	0	50
Model Connection	2	0	0	2
Validity Reporting	10	0	0	10
Exception Reporting	10	0	0	10
Final Report Output	4	0	0	4
Version Control	2	0	0	2

8.2 User Acceptance Testing

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the University Admit Eligibility Predictor project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	12	3	1	4	20
Duplicate	1	1	2	0	4
External	2	3	0	1	6
Fixed	15	3	2	17	37
Not Reproduced	1	0	0	0	1
Skipped	0	0	1	1	2
Won't Fix	0	4	3	1	8
Totals	31	14	9	24	77

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	50	0	0	50
Model Connection	2	0	0	2
Validity Reporting	10	0	0	10
Exception Reporting	10	0	0	10

Final Report Output	4	0	0	4
Version Control	2	0	0	2

9. Results

9.1 Performance Metrics

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Screenshot / Values
1.	Dashboard design	No of Visulizations / Graphs - 9
2.	Data Responsiveness	Given data has been checked for responsiveness 15 times and all data validation testing has been done
3.	Amount Data to Rendered (DB2 Metrics)	9 data points
4.	Utilization of Data Filters	Data preprocessing has been done 6 data filters have been used
5.	Effective User Story	No of Scene Added - 10
6.	Descriptive Reports	No of Visulizations / Graphs - 8

10.Advantages and Disadvantages

ADVANTAGES:

- User Friendly – The UI of the application is highly user friendly and the navigation in the website is smooth.
- Fast Results – The response time of the application is negligible and makes the whole plasma donation process hassle-free.

- Speed – This website is fast and offers great accuracy as compared to manual registered keeping.
- Maintenance – This website runs with relatively low maintenance under IBM Cloud.

DISADVANTAGES:

- Vulnerable – This application is vulnerable to the downtime of IBM Cloud and Database.
- Fake Information – This application cannot distinguish fake information.
- Auto Verification – This application does not have the facility of auto verifying genuine users.
- Internet – The working of the application requires an established internet connection.

11.Conclusion

An efficient and user-friendly way of predicting the chance of admission is implemented using the University Admit Eligibility Predictor website deployed on the IBM Cloud Platform.

To ensure smooth functioning of the website operation, it has been deployed on IBM Watson & User Interface is designed using Flask.

Machine Learning model is used to predict the Chance of admission.

12.Future Scope

Upgrading to a smoother user interface helps users understand how our website works. It will definitely help more users predict their chances of admission.

Using an Elastic Load Balancer can also help you handle excessive numbers of concurrent users and keep your website available with negligible downtime.

13.Appendix

Source code:

UniversityAdmitPredictor.ipynb

```
##Importing necessary header files
```

In [118]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [119]:

```
#Data Reading and analysis
```

In [120]:

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes
your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='V_5EQADphmCBJ_7pZlCDNapB1MvUskv-Sfnoo0-SRpgG',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'uniadmit-donotdelete-pr-6xc3ejhu53roqc'
object_key = 'Admission_Predict.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body
)

data = pd.read_csv(body)
data.head()
```

Out[120]:

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65

In [121]:

```
data.drop(["Serial No."], axis=1, inplace=True)
```

In [122]:

```
data.head()
```

Out[122]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65

In [123]:

```
data.describe()
```

Out[123]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000
mean	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0.547500	0.724350
std	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0.498362	0.142609
min	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0.000000	0.340000
25%	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0.000000	0.640000

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
50%	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1.000000	0.730000
75%	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1.000000	0.830000
max	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1.000000	0.970000

In [124]:

```
data.info()
RangeIndex: 400 entries, 0 to 399
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   GRE Score          400 non-null    int64
 1   TOEFL Score        400 non-null    int64
 2   University Rating  400 non-null    int64
 3   SOP                400 non-null    float64
 4   LOR                400 non-null    float64
 5   CGPA               400 non-null    float64
 6   Research           400 non-null    int64
 7   Chance of Admit    400 non-null    float64
dtypes: float64(4), int64(4)
memory usage: 25.1 KB
```

In [125]:

```
data.isnull().sum()
```

Out[125]:

```
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
Chance of Admit      0
dtype: int64
```

In [126]:

```
plt.scatter(data['GRE Score'],data['CGPA'])
plt.title('CGPA vs GRE Score')
plt.xlabel('GRE Score')
plt.ylabel('CGPA')
plt.show()
```

In [127]:

```
plt.scatter(data['CGPA'],data['SOP'])
plt.title('SOP for CGPA')
plt.xlabel('CGPA')
plt.ylabel('SOP')
plt.show()
```

In [128]:

```
data[data.CGPA >= 8.5].plot(kind='scatter', x='GRE Score', y='TOEFL
Score',color="BLUE")

plt.xlabel("GRE Score")
plt.ylabel("TOEFL SCORE")
plt.title("CGPA>=8.5")
plt.grid(True)

plt.show()
```

In [129]:

```
data["GRE Score"].plot(kind = 'hist',bins = 200,figsize = (6,6))

plt.title("GRE Scores")
plt.xlabel("GRE Score")
plt.ylabel("Frequency")

plt.show()
```

In [130]:

```
p = np.array([data["TOEFL Score"].min(),data["TOEFL Score"].mean(),data["TOEFL
Score"].max()])
r = ["Worst","Average","Best"]
plt.bar(p,r)

plt.title("TOEFL Scores")
plt.xlabel("Level")
plt.ylabel("TOEFL Score")

plt.show()
```

In [131]:

```
g = np.array([data["GRE Score"].min(),data["GRE Score"].mean(),data["GRE
Score"].max()])
h = ["Worst","Average","Best"]
plt.bar(g,h)

plt.title("GRE Scores")
plt.xlabel("Level")
plt.ylabel("GRE Score")

plt.show()
```

In [132]:

```
plt.figure(figsize=(10, 10))

sns.heatmap(data.corr(), annot=True, linewidths=0.05, fmt= '.2f',cmap="magma")

plt.show()
```

In [133]:

```
data.Research.value_counts()

sns.countplot(x="University Rating",data=data)
```

Out[133]:

In [134]:
```
sns.barplot(x="University Rating", y="Chance of Admit ", data=data)
```
Out[134]:

In [135]:
```
#Train-test split
```
In [136]:
```
X=data.drop(['Chance of Admit '],axis=1) #input data_set
y=data['Chance of Admit '] #output labels
```
In [137]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15)
```
In [138]:
```
#Modelling and training
```
In [139]:
```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
X_train[X_train.columns] = scaler.fit_transform(X_train[X_train.columns])
X_test[X_test.columns] = scaler.transform(X_test[X_test.columns])
X_train.head()
```
Out[139]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
114	0.42	0.464286	0.50	0.625	0.500	0.461255	1.0
171	0.88	0.892857	1.00	0.750	0.875	0.690037	1.0
322	0.48	0.535714	0.25	0.375	0.750	0.394834	0.0
291	0.20	0.357143	0.25	0.125	0.250	0.247232	0.0
148	0.98	0.857143	0.75	0.750	0.625	0.959410	1.0

In [140]:
```
from sklearn.ensemble import GradientBoostingRegressor
rgr = GradientBoostingRegressor()
rgr.fit(X_train,y_train)
```
Out[140]:
```
GradientBoostingRegressor()
```
In [141]:
```
rgr.score(X_test,y_test)
```
Out[141]:
```
0.6734968747397405
```
In [142]:
```
y_predict=rgr.predict(X_test)
```
In [143]:
```
from sklearn.metrics import mean_squared_error, r2_score,mean_absolute_error
import numpy as np
print('Mean Absolute Error:', mean_absolute_error(y_test, y_predict))
```

```
print('Mean Squared Error:', mean_squared_error(y_test, y_predict))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_predict)))
```

```
Mean Absolute Error: 0.06339105941564477
Mean Squared Error: 0.007488349177844049
Root Mean Squared Error: 0.08653524818155922
```

In [144]:

```
y_train = (y_train>0.5)
y_test = (y_test>0.5)
```

In [145]:

```
from sklearn.linear_model._logistic import LogisticRegression

lore = LogisticRegression(random_state=0, max_iter=1000)

lr = lore.fit(X_train, y_train)
```

In [146]:

```
y_pred = lr.predict(X_test)
```

In [147]:

```
#Performance Metrics
```

In [148]:

```
from sklearn.metrics import accuracy_score, recall_score, roc_auc_score,
confusion_matrix

print('Accuracy Score:', accuracy_score(y_test, y_pred))
print('Recall Score:', recall_score(y_test, y_pred))
print('ROC AUC Score:', roc_auc_score(y_test, y_pred))
print('Confusion Matrix:\n', confusion_matrix(y_test, y_pred))
```

```
Accuracy Score: 0.85
Recall Score: 1.0
ROC AUC Score: 0.5
Confusion Matrix:
 [[ 0  9]
 [ 0 51]]
```

In [149]:

```
#IBM Deployment
```

In [150]:

```
!pip install -U ibm-watson-machine-learning
```

```
Requirement already satisfied: ibm-watson-machine-learning in /opt/conda/envs/Pytho
n-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib
/python3.9/site-packages (from ibm-watson-machine-learning) (4.8.2)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/s
ite-packages (from ibm-watson-machine-learning) (0.3.3)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.
9/site-packages (from ibm-watson-machine-learning) (21.3)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/
site-packages (from ibm-watson-machine-learning) (1.26.7)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/
lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.3.4)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/li
b/python3.9/site-packages (from ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/
site-packages (from ibm-watson-machine-learning) (2022.9.24)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9
/site-packages (from ibm-watson-machine-learning) (2.26.0)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9
/site-packages (from ibm-watson-machine-learning) (0.8.9)
```

```
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Py
thon-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-
learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3
.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learni
ng) (2.11.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9
/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learning
) (0.10.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Pytho
n-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11
.*->ibm-watson-machine-learning) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/pytho
n3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-watson-machine-learning) (2021.
3)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/pyth
on3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-watson-machine-learning) (1.20
.3)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9
/site-packages (from python-dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.11.0->ibm-cos
-sdk==2.11.*->ibm-watson-machine-learning) (1.15.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/pytho
n3.9/site-packages (from requests->ibm-watson-machine-learning) (3.3)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from requests->ibm-watson-machine-learning) (2.0.4
)
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-3.9/lib/python3.
9/site-packages (from importlib-metadata->ibm-watson-machine-learning) (3.6.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3
.9/lib/python3.9/site-packages (from packaging->ibm-watson-machine-learning) (3.0.4
)
```

In [151]:

```
from ibm_watson_machine_learning import APIClient
import json
```

In [152]:

```
#Authenticate and set space
```

In [153]:

```
wml_credentials = {
    "apikey":"_jEZ9faTKtRDlZDFHcrrAbuQFdvxbBeiu8W1D8-4z7ng",
    "url":"https://us-south.ml.cloud.ibm.com"
}
```

In [154]:

```
wml_client = APIClient(wml_credentials)
```

In [155]:

```
wml_client.spaces.list()
Note: 'limit' is not provided. Only first 50 records will be displayed if the numbe
r of records exceed 50
------------------------------------  ------------------  ------------------------
ID                                    NAME                CREATED
85ccaf78-6115-4614-b3ad-9b389883e43d  UniversityPredictor  2022-11-18T08:48:38.039Z
------------------------------------  ------------------  ------------------------
```

In [156]:

```
SPACE_ID= "85ccaf78-6115-4614-b3ad-9b389883e43d"
```

In [157]:

```
wml_client.set.default_space(SPACE_ID)
```

Out[157]:

```
'SUCCESS'
```

In [158]:

```
wml_client.software_specifications.list(100)
```

```
------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7  base
autoai-kb_rt22.2-py3.10         125b6d9a-5b1f-5e8d-972a-b251688ccf40  base
runtime-22.1-py3.9              12b83a17-24d8-5082-900f-0ab31fbfd3cb  base
scikit-learn_0.22-py3.6         154010fa-5b3b-4ac1-82af-4d5ee5abbc85  base
default_r3.6                    1b70aec3-ab34-4b87-8aa0-a4a3c8296a36  base
pytorch-onnx_1.3-py3.6          1bc6029a-cc97-56da-b8e0-39c3880dbbe7  base
kernel-spark3.3-r3.6            1c9e5454-f216-59dd-a20e-474a5cdf5988  base
pytorch-onnx_rt22.1-py3.9-edt   1d362186-7ad5-5b59-8b6c-9d0880bde37f  base
tensorflow_2.1-py3.6            1eb25b84-d6ed-5dde-b6a5-3fbdf1665666  base
spark-mllib_3.2                 20047f72-0a98-58c7-9ff5-a77b012eb8f5  base
tensorflow_2.4-py3.8-horovod    217c16f6-178f-56bf-824a-b19f20564c49  base
runtime-22.1-py3.9-cuda         26215f05-08c3-5a41-a1b0-da66306ce658  base
do_py3.8                        295addb5-9ef9-547e-9bf4-92ae3563e720  base
autoai-ts_3.8-py3.8             2aa0c932-798f-5ae9-abd6-15e0c2402fb5  base
tensorflow_1.15-py3.6           2b73a275-7cbf-420b-a912-eae7f436e0bc  base
kernel-spark3.3-py3.9           2b7961e2-e3b1-5a8c-a491-482c8368839a  base
pytorch_1.2-py3.6               2c8ef57d-2687-4b7d-acce-01f94976dac1  base
spark-mllib_2.3                 2e51f700-bca0-4b0d-88dc-5c6791338875  base
pytorch-onnx_1.1-py3.6-edt      32983cea-3f32-4400-8965-dde874a8d67e  base
spark-mllib_3.0-py37            36507ebe-8770-55ba-ab2a-eafe787600e9  base
spark-mllib_2.4                 390d21f8-e58b-4fac-9c55-d7ceda621326  base
autoai-ts_rt22.2-py3.10         396b2e83-0953-5b86-9a55-7ce1628a406f  base
xgboost_0.82-py3.6              39e31acd-5f30-41dc-ae44-60233c80306e  base
pytorch-onnx_1.2-py3.6-edt      40589d0e-7019-4e28-8daa-fb03b6f4fe12  base
pytorch-onnx_rt22.2-py3.10      40e73f55-783a-5535-b3fa-0c8b94291431  base
default_r36py38                 41c247d3-45f8-5a71-b065-8580229facf0  base
autoai-ts_rt22.1-py3.9          4269d26e-07ba-5d40-8f66-2d495b0c71f7  base
autoai-obm_3.0                  42b92e18-d9ab-567f-988a-4240ba1ed5f7  base
pmml-3.0_4.3                    493bcb95-16f1-5bc5-bee8-81b8af80e9c7  base
spark-mllib_2.4-r_3.6           49403dff-92e9-4c87-a3d7-a42d0021c095  base
xgboost_0.90-py3.6              4ff8d6c2-1343-4c18-85e1-689c965304d3  base
pytorch-onnx_1.1-py3.6          50f95b2a-bc16-43bb-bc94-b0bed208c60b  base
autoai-ts_3.9-py3.8             52c57136-80fa-572e-8728-a5e7cbb42cde  base
spark-mllib_2.4-scala_2.11      55a70f99-7320-4be5-9fb9-9edb5a443af5  base
spark-mllib_3.0                 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9  base
autoai-obm_2.0                  5c2e37fa-80b8-5e77-840f-d912469614ee  base
spss-modeler_18.1               5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b  base
cuda-py3.8                      5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e  base
runtime-22.2-py3.10-xc          5e8cddff-db4a-5a6a-b8aa-2d4af9864dab  base
autoai-kb_3.1-py3.7             632d4b22-10aa-5180-88f0-f52dfb6444d7  base
pytorch-onnx_1.7-py3.8          634d3cdc-b562-5bf9-a2d4-ea90a478456b  base
spark-mllib_2.3-r_3.6           6586b9e3-ccd6-4f92-900f-0f8cb2bd6f0c  base
```

```
tensorflow_2.4-py3.7             65e171d7-72d1-55d9-8ebb-f813d620c9bb  base
spss-modeler_18.2                687eddc9-028a-4117-b9dd-e57b36f1efa5  base
pytorch-onnx_1.2-py3.6           692a6a4d-2c4d-45ff-a1ed-b167ee55469a  base
spark-mllib_2.3-scala_2.11       7963efe5-bbec-417e-92cf-0574e21b4e8d  base
spark-mllib_2.4-py37             7abc992b-b685-532b-a122-a396a3cdbaab  base
caffe_1.0-py3.6                  7bb3dbe2-da6e-4145-918d-b6d84aa93b6b  base
pytorch-onnx_1.7-py3.7           812c6631-42b7-5613-982b-02098e6c909c  base
cuda-py3.6                       82c79ece-4d12-40e6-8787-a7b9e0f62770  base
tensorflow_1.15-py3.6-horovod    8964680e-d5e4-5bb8-919b-8342c6c0dfd8  base
hybrid_0.1                       8c1a58c6-62b5-4dc4-987a-df751c2756b6  base
pytorch-onnx_1.3-py3.7           8d5d8a87-a912-54cf-81ec-3914adaa988d  base
caffe-ibm_1.0-py3.6              8d863266-7927-4d1e-97d7-56a7f4c0a19b  base
runtime-22.2-py3.10-cuda         8ef391e4-ef58-5d46-b078-a82c211c1058  base
spss-modeler_17.1                902d0051-84bd-4af6-ab6b-8f6aa6fdeabb  base
do_12.10                         9100fd72-8159-4eb9-8a0b-a87e12eefa36  base
do_py3.7                         9447fa8b-2051-4d24-9eef-5acb0e3c59f8  base
spark-mllib_3.0-r_3.6            94bb6052-c837-589d-83f1-f4142f219e32  base
cuda-py3.7-opence                94e9652b-7f2d-59d5-ba5a-23a414ea488f  base
nlp-py3.8                        96e60351-99d4-5a1c-9cc0-473ac1b5a864  base
cuda-py3.7                       9a44990c-1aa1-4c7d-baf8-c4099011741c  base
hybrid_0.2                       9b3f9040-9cee-4ead-8d7a-780600f542f7  base
spark-mllib_3.0-py38             9f7a8fc1-4d3c-5e65-ab90-41fa8de2d418  base
autoai-kb_3.3-py3.7              a545cca3-02df-5c61-9e88-998b09dc79af  base
spark-mllib_3.0-py39             a6082a27-5acc-5163-b02c-6b96916eb5e0  base
runtime-22.1-py3.9-do            a7e7dbf1-1d03-5544-994d-e5ec845ce99a  base
default_py3.8                    ab9e1b80-f2ce-592c-a7d2-4f2344f77194  base
tensorflow_rt22.1-py3.9          acd9c798-6974-5d2f-a657-ce06e986df4d  base
kernel-spark3.2-py3.9            ad7033ee-794e-58cf-812e-a95f4b64b207  base
autoai-obm_2.0 with Spark 3.0    af10f35f-69fa-5d66-9bf5-acb58434263a  base
runtime-22.2-py3.10              b56101f1-309d-549b-a849-eaa63f77b2fb  base
default_py3.7_opence             c2057dd4-f42c-5f77-a02f-72bdbd3282c9  base
tensorflow_2.1-py3.7             c4032338-2a40-500a-beef-b01ab2667e27  base
do_py3.7_opence                  cc8f8976-b74a-551a-bb66-6377f8d865b4  base
spark-mllib_3.3                  d11f2434-4fc7-58b7-8a62-755da64fdaf8  base
autoai-kb_3.0-py3.6              d139f196-e04b-5d8b-9140-9a10ca1fa91a  base
spark-mllib_3.0-py36             d82546d5-dd78-5fbb-9131-2ec309bc56ed  base
autoai-kb_3.4-py3.8              da9b39c3-758c-5a4f-9cfd-457dd4d8c395  base
kernel-spark3.2-r3.6             db2fe4d6-d641-5d05-9972-73c654c60e0a  base
autoai-kb_rt22.1-py3.9           db6afe93-665f-5910-b117-d879897404d9  base
tensorflow_rt22.1-py3.9-horovod  dda170cc-ca67-5da7-9b7a-cf84c6987fae  base
autoai-ts_1.0-py3.7              deef04f0-0c42-5147-9711-89f9904299db  base
tensorflow_2.1-py3.7-horovod     e384fce5-fdd1-53f8-bc71-11326c9c635f  base
default_py3.7                    e4429883-c883-42b6-87a8-f419d64088cd  base
do_22.1                          e51999ba-6452-5f1f-8287-17228b88b652  base
autoai-obm_3.2                   eae86aab-da30-5229-a6a6-1d0d4e368983  base
runtime-22.2-r4.2                ec0a3d28-08f7-556c-9674-ca7c2dba30bd  base
tensorflow_rt22.2-py3.10         f65bd165-f057-55de-b5cb-f97cf2c0f393  base
do_20.1                          f686cdd9-7904-5f9d-a732-01b0d6b10dc5  base
-------------------------------  ------------------------------------  ----
```

In [159]:

```
import sklearn
sklearn.__version__
```

Out[159]:

```
'1.0.2'
```

In [160]:

```
MODEL_NAME = 'UniPredictorJupyter'
```

```
DEPLOYMENT_NAME = 'UniversityPredictor'
DEMO_MODEL = lr
```

In [161]:

```
# Set Python Version
software_spec_uid = wml_client.software_specifications.get_id_by_name('runtime-
22.1-py3.9')
```

In [162]:

```
# Setup model meta
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn_1.0',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

In [163]:

```
#Save model

model_details = wml_client.repository.store_model(
    model=DEMO_MODEL,
    meta_props=model_props,
    training_data=X_train,
    training_target=y_train
)
```

In [164]:

```
model_details
```

Out[164]:


```
{'entity': {'hybrid_pipeline_software_specs': [],
  'label_column': 'Chance of Admit ',
  'schemas': {'input': [{'fields': [{'name': 'GRE Score', 'type': 'float64'},
      {'name': 'TOEFL Score', 'type': 'float64'},
      {'name': 'University Rating', 'type': 'float64'},
      {'name': 'SOP', 'type': 'float64'},
      {'name': 'LOR ', 'type': 'float64'},
      {'name': 'CGPA', 'type': 'float64'},
      {'name': 'Research', 'type': 'float64'}],
     'id': '1',
     'type': 'struct'}],
   'output': []},
  'software_spec': {'id': '12b83a17-24d8-5082-900f-0ab31fbfd3cb',
   'name': 'runtime-22.1-py3.9'},
  'type': 'scikit-learn_1.0'},
 'metadata': {'created_at': '2022-11-18T13:54:15.992Z',
  'id': 'e025bab3-1cee-46fe-a3c5-496d408eb8c2',
  'modified_at': '2022-11-18T13:54:18.680Z',
  'name': 'UniPredictorJupyter',
  'owner': 'IBMid-667000FKLK',
  'resource_key': '8eb0c55a-2f23-4034-8caf-8e6deb5434fe',
  'space_id': '85ccaf78-6115-4614-b3ad-9b389883e43d'},
 'system': {'warnings': []}}
```


In [165]:

```
model_id = wml_client.repository.get_model_id(model_details)
model_id
```

Out[165]:


```
'e025bab3-1cee-46fe-a3c5-496d408eb8c2'
```


In [166]:

```
#X_train
```

In [167]:

```
#from sklearn.linear_model import LinearRegression
multiple_lin_reg = LinearRegression()
multiple_lin_reg.fit(X_train,y_train)

y_pred_mlr = multiple_lin_reg.predict(X_test)

r2_score_mlr = r2_score(y_test,y_pred_mlr)
print("Mutiple Linear Regression's Score = {:.3f}".format(r2_score_mlr))
```

```
Mutiple Linear Regression's Score = 0.258
```

In [168]:

```
#multiple_lin_reg.predict(X_train)
```

In [169]:

```
# Set meta
deployment_props = {
    wml_client.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,
    wml_client.deployments.ConfigurationMetaNames.ONLINE: {}
}
```

In [170]:

```
# Deploy
deployment = wml_client.deployments.create(
    artifact_uid=model_id,
    meta_props=deployment_props
)
```

```
#########################################################################################
####

Synchronous deployment creation for uid: 'e025bab3-1cee-46fe-a3c5-496d408eb8c2' sta
rted

#########################################################################################
####


initializing
Note: online_url is deprecated and will be removed in a future release. Use serving
_urls instead.

ready


------------------------------------------------------------------------------------
-------------
Successfully finished deployment creation, deployment_uid='04d68702-0270-4707-b4e1-
505e1f90de26'
------------------------------------------------------------------------------------
-------------
```

Main Page.html

```
<!DOCTYPE html>
<html>
<head>
<title>University Admit Eligibility Predictor</title>
<link rel = "icon" href = "/static/img/Logo.jpg" type = "image/x-icon">
<body style=" background-image:url('/static/img/background 1.jpg');background-repeat:
no-repeat;background-attachment:fixed;background-size: 100% 100%;" >
<div style="position:absolute;left:1300px">
<img src="/static/img/Logo.jpg" alt="Website Logo" style="width:40px;height:40px;">
</div>
<div style="position:absolute;top:20px;left:280px">
<p style="font-family:serif;font-size:45px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<strong>University Admit Eligibility Predictor</strong>
</p>
</div>
<div style="position:absolute;top:150px;left:220px">
<p style="font-family:serif;font-size:100px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<p style="color:#F6E9F9"><i>Students are often worried about their chances of
admission to University. The aim of this project is to help students in shortlisting
universities with their profiles. The predicted output gives them a fair idea about
their admission chances in a particular university. This analysis should also help
students who are currently preparing or will be preparing to get a better
idea.</i></p>
</p>
</div>
<div style="position:absolute;top:450px;left:470px">
<p style="font-family:serif;font-size:30px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<a style="color:#F6E9F9" href="/Page 1" target="_blank"><i>Proceed to check
Eligibility</i></a>
</p>
</div>
</body>
</html>
```

Page 1.html

```
<!DOCTYPE html>
<html>
<head>
<title>University Admit Eligibility Predictor</title>
<link rel = "icon" href = "/static/img/Logo.jpg" type = "image/x-icon">
<!--<link rel="stylesheet" href="style.css">-->
```

```
<style>
h1{
font-family:Verdana;
font-size:1.7rem;
color:#d77fa1;
text-align:center;
}
#div1{
padding-left:400px;
}
.nameip
{
background-color: black;
color: White ;
}
.draglist{
width: max-content;
height: max-content;
color: black;
background-color:#F3E9DD;
border: 1px solid orange ;
border-radius: 5%;
margin: 5px;
}
input,select{
margin: 10px;
}
</style>

</head>
<body style=" background-image: url('/static/img/background.jpg' ); background-
repeat:no-repeat;background-attachment: fixed;background-size: 100% 100%;" ></body>
<br><br>
<header>
<div style="position:absolute;left:1310px;top:0">
<img src="/static/img/Logo.jpg" alt="Website Logo"
style="width:40px;height:40px;">
</div>
</header>
<h1
style="font-family:serif;font-size:40px;text-align:center;text-
decoration:underline;color: white">UNIVERSITY ADMIT ELIGIBILITY PREDICTOR<br><br></h1>
<div id="div1">
<form id="warship" method="post" action="."
style="position:absolute;top:140px;left:350px;font-family:serif;font-
size:20px;color:black">
<label style="margin-top:10px;" for="name">Name : </label>
<input type="text" id="name" name="name" onfocus="changeBorder(this)"
```

```
onblur="reverseChange(this)" pattern="[a-zA-Z]+" oninvalid="alert('Enter ValidName')"
required/><br />
<label for="GRE Score">GRE Score : </label>
<input type="text" id="GRE Score" name="GRE Score" onfocus="changeBorder(this)"><br>
<label for="TOEFL Score">TOEFL Score : </label>
<input type="text" id="TOEFL Score" name="TOEFL Score"
onfocus="changeBorder(this)"><br>
<label for="University Rating">University Rating : </label>
<select id="University Rating" name="University Rating" required>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
</select><br/>
<label for="SOP">SOP : </label>
<select id="SOP" name="SOP" required>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
</select><br/>
<label for="LOR">LOR : </label>
<select id="LOR" name="LOR" required>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
</select><br/>
<label for="CGPA">CGPA : </label>
<input type="text" id="CGPA" name="CGPA" onfocus="changeBorder(this)"><br>
<label for="Research">Research : </label>
<select id="Research" name="Research" required>
<option value="0">0</option>
<option value="1">1</option>
</select><br/>
<input type="submit" value="Submit">
<input type="reset">
</form></div>
</body>
</html>
```

chance.html

```
<!DOCTYPE html>
<html>
<head>
<title>University Admit Eligibility Predictor</title>
<link rel = "icon" href = "/static/img/Logo.jpg" type = "image/x-icon">
<body style=" background-image:url('/static/img/background 1.jpg');background-repeat:
no-repeat;background-attachment:fixed;background-size: 100% 100%;" >
<div style="position:absolute;left:1300px">
<img src="/static/img/Logo.jpg" alt="Website Logo" style="width:40px;height:40px;">
</div>
<div style="position:absolute;top:20px;left:280px">
<p style="font-family:serif;font-size:45px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<strong>University Admit Eligibility Predictor</strong>
<div style="position:absolute;top:70px;left:200px">
<p style="font-family:serif;font-size:80px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<p style="color: black"><i><strong>You have a good chance of admit!
Congratulations!</strong><br>{{content}}%</i></p>

</p>
</div>
</body>
</html>
```

nochance.html

```
<!DOCTYPE html>
<html>
<head>
<title>University Admit Eligibility Predictor</title>
<link rel = "icon" href = "/static/img/Logo.jpg" type = "image/x-icon">
<body style=" background-image:url('/static/img/background 1.jpg');background-repeat:
no-repeat;background-attachment:fixed;background-size: 100% 100%;" >
<div style="position:absolute;left:1300px">
<img src="/static/img/Logo.jpg" alt="Website Logo" style="width:40px;height:40px;">
</div>
<div style="position:absolute;top:20px;left:280px">
<p style="font-family:serif;font-size:45px;text-align:center;text-
decoration:underline;color:#F6E9F9">
<strong>University Admit Eligibility Predictor</strong>
<div style="position:absolute;top:70px;left:200px">
<p style="font-family:serif;font-size:80px;text-align:center;text-
decoration:underline;color:#F6E9F9">
```

```
<p style="color: black"><i><strong>You do not have a good chance of admit!
Sorry!</strong><br>{{content}}%</i></p>

</p>
</div>
</body>
</html>
```

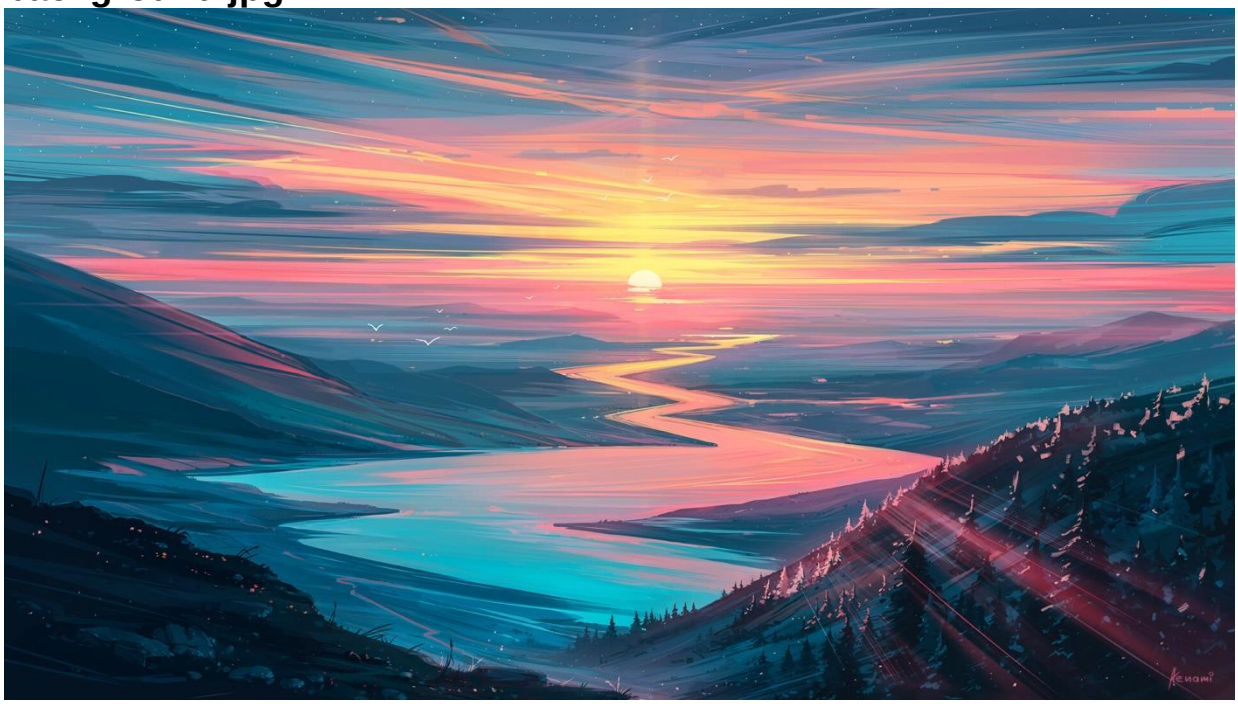
app.py

```
from flask import Flask, render_template, redirect, url_for, request,jsonify
import requests

API_KEY = "_jEZ9faTKtRDlZDFHcrrAbuQFdvxbBeiu8W1D8-4z7ng"
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__)

@app.route("/")
def home():
    return render_template("Main Page.html")


@app.route("/Page 1",methods = ['POST','GET'])
def input():
    return render_template("Page 1.html")

@app.route("/", methods = ['POST','GET'])
def prediction():
    if request.method == 'POST':
        arr = []
        for i in ["GRE Score","TOEFL Score","University
Rating","SOP","LOR","CGPA","Research"]:
            val = request.form[i]
            if val == '':
                return redirect(url_for("Page 1.html"))
            arr.append(float(val))
        #print(arr)
        # deepcode ignore HardcodedNonCryptoSecret: <please specify a reason of
ignoring this>

        payload_scoring = {
            "input_data": [{"fields":[  'GRE Score',
                                        'TOEFL Score',
                                        'University Rating',
                                        'SOP',
                                        'LOR ',
                                        'CGPA',
                                        'Research'],
```

```
                            "values": [arr]
                            }]
                        }

        response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/ed4ff532-801c-457b-85b9-
ffa103d3a064/predictions?version=2022-11-18',
json=payload_scoring,headers=header).json()
        print(response_scoring)
        result = response_scoring['predictions'][0]['values']
        print(result)

        if result[0][0] > 0.5:
            return redirect(url_for('chance', percent=result[0][0]*100))
        else:
            return redirect(url_for('no_chance', percent=result[0][0]*100))
    else:
        return redirect(url_for("Page 1"))
    return ""




@app.route("/chance/<percent>")
def chance(percent):
    return render_template("chance.html", content=[percent])

@app.route("/nochance/<percent>")
def no_chance(percent):
    return render_template("noChance.html", content=[percent])



@app.route('/<path:path>')
def catch_all(path):
    return render_template("Page 1.html")

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

Images used in the html pages

background.jpg



background1.jpg

logo.jpg

GIT Link : https://github.com/IBM-EPBL/IBM-Project-17372-1659636785

Project Demo Link : https://drive.google.com/file/d/1HEBNs7B-pCA2mftMgsjLqepSI_O3j8yK/view?usp=sharing