UNIVERSITY ADMIT ELIGIBILITY PREDICTOR

NAALAIYA THIRAN PROJECT BASED LEARNING ON PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

A PROJECT REPORT

Submitted by

AARTHI .M **(611819104001)**

GAYATHRI .E **(611819104016)**

SHAMRITHA .P.R **(611819104042)**

SHAFIYA AFSAN .Z **(611819104041)**

TEAM ID: PNT2022TMID40873

FACULTY MENTOR NAME: C.PRAKASH NARAYANAN

INDUSTRY MENTOR NAME: SANDESH.P

EVALUATOR NAME: B.NEELU

P.S.V COLLEGE OF ENGINEERING AND TECHNOLOGY

(An ISO 9001:2015 Certified Institution)

(Accredited by NAAC with 'A' Grade)

KRISHNAGIRI-635108

NOVEMBER, 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled "**UNIVERSITY ADMIT ELIGIBILITY PREDICTOR"** is the bonafide work of **"AARTHI .M – (611819104001) , GAYATHRI .E – (611819104016) , SHAMRITHA .P.R – (611819104042)** and **SHAFIYA AFSAN .Z –(611819104041)"** who carried out there search under my supervision .

SIGNATURE
HEAD OF THE DEPARTMENT

Prof.B.SAKTHIVEL., M.E., (Ph.D).,
Dept. of Computer Science & Engineering,
P.S.V College of Engineering
and Technology,
Krishnagiri (D.T), 635108.

SIGNATURE
FACULTY MENTOR

Prof.C.PRAKASH NARAYANAN., M.E.
Dept. of Computer Science & Engineering,
P.S.V College of Engineering
and Technology,
Krishnagiri (D.T), 635 108.

Submitted for the **Naalaiya thiran Project based learning on professional readiness for innovation, Employability and Entrepreneurship** _____ at P.S.V College of Engineering and Technology, Krishnagiri.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

At this pleasing moment of having successfully completed my Project, I wish to convey our sincere thanks and gratitude to the management of our college and our beloved Chairman, **Dr.P.SELVAM M.A.,B.Ed.,Ph.D.,D.Litt,** who provided all the facilities to me.

I would like to express my sincere thanks to my beloved Principal **Dr.P.LAWRENCE M.E.,Ph.D.**, for forwarding us to do our project and offering adequate duration in completing my project.

I also express my sincere thanks to **Prof.B.SAKTHIVEL M.E.,(Ph.D).,**Head of the Department of Computer Science and Engineering for providing all the facilities in the successful completion of my project.

I have great pleasure to express my sense of gratitude to my internal guide **Prof.C.PRAKASH NARAYANAN M.E.,** Assistant Professor, whose guidance and encouragement made this project an interesting educational experience.

AARTHI.M
GAYATHRI.E
SHAMRITHA.P.R
SHAFIYA AFSAN.Z

CONTENT

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW:

Students are often worried about their chances of admission to university. The aim of this project is to help students in short listing 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. Student admission problem is very important in educational institutions. This project addresses machine learning models to predict the chance of a student to be admitted to a university. This will assist students to know in advance if they have a chance to get accepted. The main purpose of this project is to manage all the processes of admission in a simple and efficient way. The University prediction would be the easiest mode to predict University / colleges a person is applicable for as well as it would be unbiased and totally transparent. Moreover applying to only that colleges /university where the student has genuine chance would even reduce application process. This project helps the students to save their time and also it will help them to limit their number of application to a small number by providing them the suggestion of the universities where they have the best chance of securing admissions thus saving more money on the application fees. The applicant or student should make their own profile which helps to predict the best suitable university for their profile.

Keywords: Chances of admission, Short listing universities, Profiles, Machine Learning, unbiased, totally transparent.

1.2 PURPOSE:

This is a Requirements Specification Document for a new web-based University Admit Eligibility Predictor. It is an AI based application that asks for the users to input their academic transcripts data and calculates their chances of admission into the University Tier that they selected. It also provides an analysis of the data and shows how chances of admissions can depend on various factors. This document describes the scope, objectives and goals of the system. In addition to describing the non-functional requirements, this document models the functional requirements with use cases, interaction diagrams and class models. This document is intended to direct the design and implementation of the target system in an object-oriented language.

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING PROBLEM:

- Numerous program and studies have been carried out on relating to University admission used many machine learning models which helps the students in the admission process to their desired University.
- Previous research done in this area used Naive Bayes algorithm which will evaluate the success probabilities of student's application into a respective University.
- Bayesian Networks Algorithm has been used to create a decision support network for evaluating the application submitted by foreign students of the University.
- This model was developed to forecast the progress of prospective students by comparing the score of students currently studying at University.
- The model does predict whether the aspiring students should be admitted to university on the basis of various scores of students.
- A Database will also be implemented for the system so that students can save their data and review and edit it as they progress with the most recent predictions being saved with their profile.
- The system will be available to all users from any location as long as they have an Internet connection.

2.2 REFERENCES:

1. Ferrari, Francesca, Raffaella Striani, Stefania Minosi, Roberto De Fazio, Paolo Visconti, Luigi Patrono, Luca Catarinucci, Carola Esposito Corcione, and Antonio Greco. "An innovative IoT-oriented prototype platform for the management and valorisation of the organic fraction of municipal solid waste." Journal of Cleaner Production 247 (2020): 119618.

2. Zhang, Abraham, V. G. Venkatesh, Yang Liu, Ming Wan, Ting Qu, and Donald Huisingh. "Barriers to smart waste management for a circular economy in China." Journal of Cleaner Production 240 (2019): 118198.

3. Ferrari, Francesca, Raffaella Striani, Paolo Visconti, Carola Esposito Corcione, and Antonio Greco. "Durability analysis of formaldehyde/solid urban waste blends." Polymers 11, no. 11 (2019): 1838.

4. Pardini, Kellow, Joel JPC Rodrigues, Sergei A. Kozlov, Neeraj Kumar, and Vasco Furtado. "IoT-based solid waste management solutions: a survey." Journal of Sensor and Actuator Networks 8, no. 1 (2019): 5.

5. Hong, Insung, Sunghoi Park, Beomseok Lee, Jaekeun Lee, Daebeom Jeong, and Sehyun Park. "IoT-based smart garbage system for efficient food waste management." The Scientific World Journal 2014 (2014).

6. Paritosh, Kunwar, Sandeep K. Kushwaha, Monika Yadav, Nidhi Pareek, Aakash Chawade, and Vivekanand Vivekanand. "Food waste to energy: an overview of sustainable approaches for food waste management and nutrient recycling." BioMed Research International 2017 (2017).

7. Verdouw, Cor, Harald Sundmaeker, Bedir Tekinerdogan, Davide Conzon, and Teodoro Montanaro. "Architecture framework of IoT based food and farm systems: A multiple case study." Computers and Electronics in Agriculture 165 (2019): 104939.

8. Esmaeilian, Behzad, Ben Wang, Kemper Lewis, Fabio Duarte, Carlo Ratti, and Sara Behdad. "The future of waste management in smart and sustainable cities: A review and concept paper." Waste management 81 (2018): 177-195.

2.3 PROBLEM STATEMENT DEFINITION:

1. The problem statement is to design a college prediction system and to provide a probabilistic insight into college administration for overall rating, cut- offs of the colleges, admission intake and preferences of students.

2. It has always been a troublesome process for students in finding the perfect university and course for their further studies.

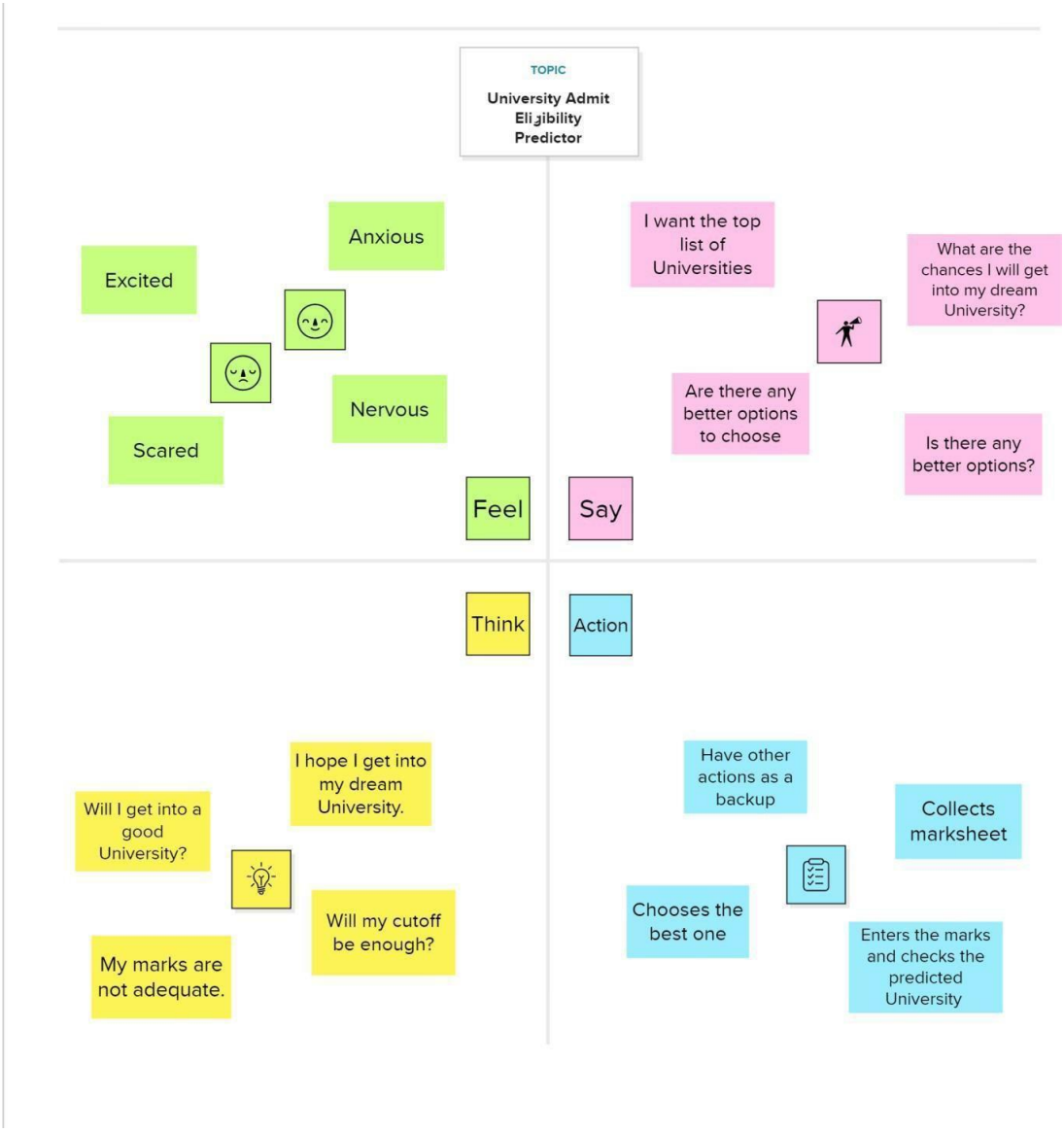
3. At times they do know which stream they want to get into, but it is not easy for them to find colleges based on their academic marks and other performances.

4. We aim to develop and provide a place which would give a probabilistic output of how likely it is to get into a university given their details.

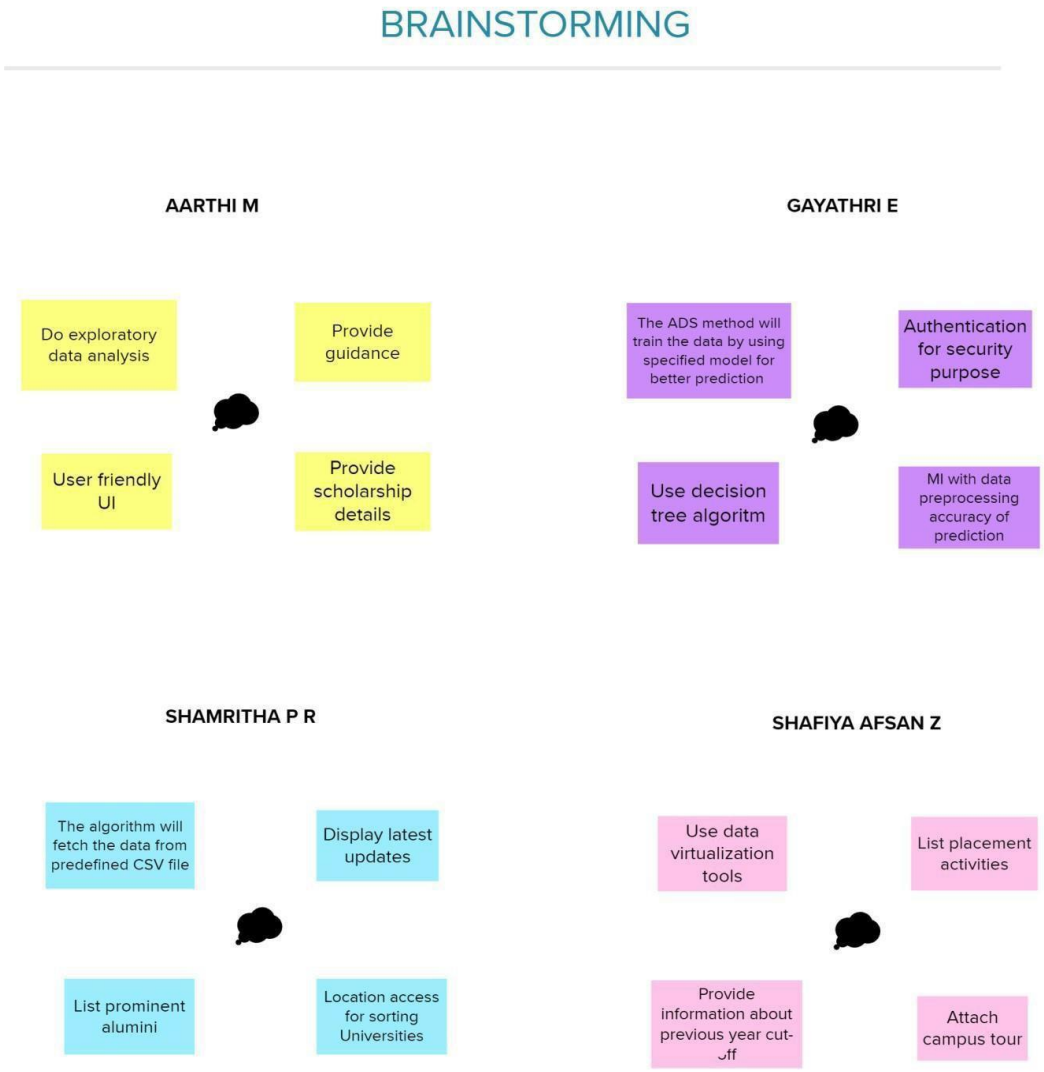
CHAPTER 3

IDEATION PHASE AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



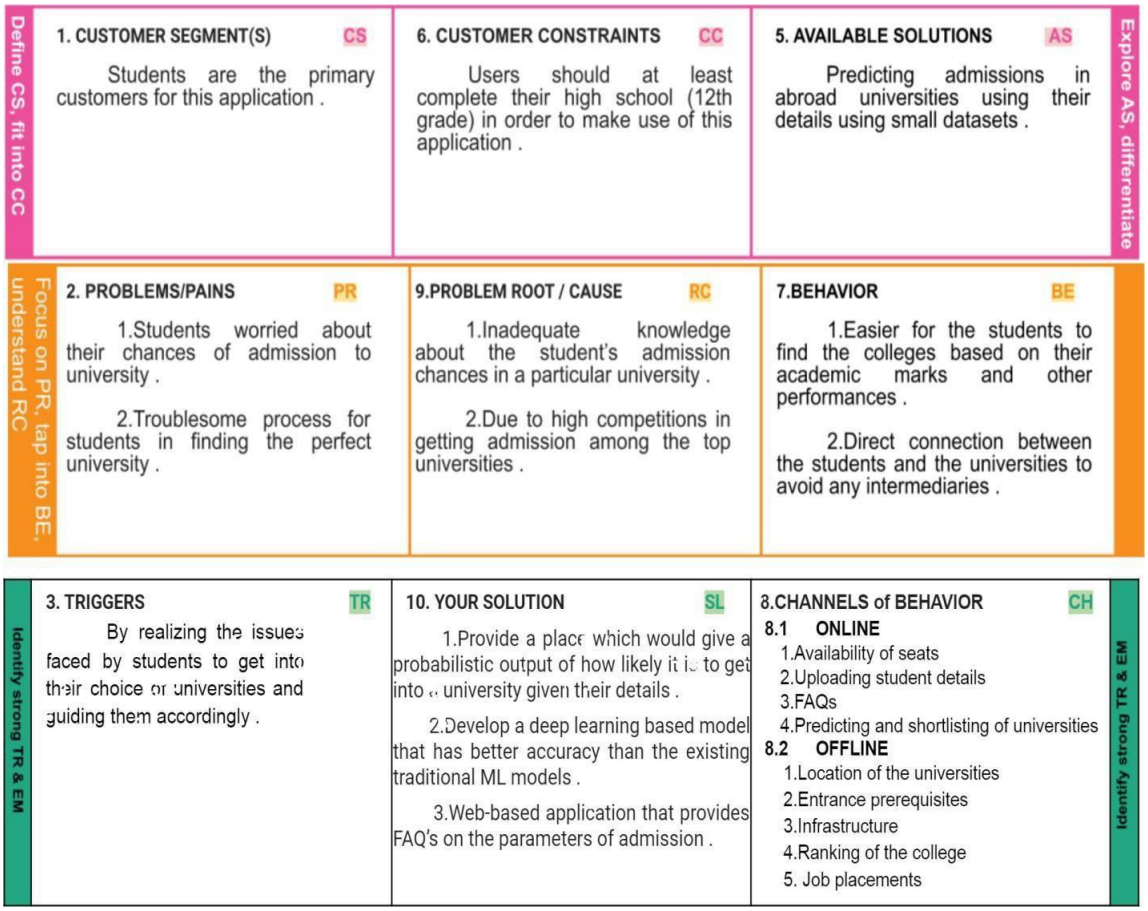
3.2 IDEATION AND BRAINSTORMING:



3.3 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Students do not have much idea about the procedures, requirement and details of the universities they want to join, so they seek help from various educational consultancies to help them secure admission in the universities based on their profile, for which the students are supposed pay a hefty amount as consultancy fee.
2.	Idea / Solution description	Providing an accurate prediction for the student's admission into the university of their choice based on various parameters like IELTS, GRE, Academic Performance, etc.
3.	Novelty / Uniqueness	It seems there are no web applications for predicting the eligibility criteria of a student for getting in to their dream university and also provide personalized insights on specific areas where they can improve.
4.	Social Impact / Customer Satisfaction	It helps student in making the right decision for choosing the universities. It cuts the cost of consultancy services by creating a direct connection between students and universities.
5.	Business Model (Revenue Model)	Universities are under immense pressure to admit more students and ensure student success. To overcome this pressure, they can make use of predictive models which help them to ease the intake process of students and improve efficiency.
6.	Scalability of the Solution	Further to reduce the immense pressure faced by the students to get admitted in a university, the model can also be evolved to consider university specific examinations and to maintain the latest eligibility criteria

3.4 PROBLEM SOLUTION FIT:



CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Profile	Complete user profile by providing the Student Academic details.
FR-2	User Search	Search for desired University based on their Academic Performance and eligibility criteria.
FR-3	User Preference	Search for Universities based on their location preference.
FR-4	Result	The list of universities is filtered based on the eligibility of the students where the order of the list will be based on the ratings of the university.

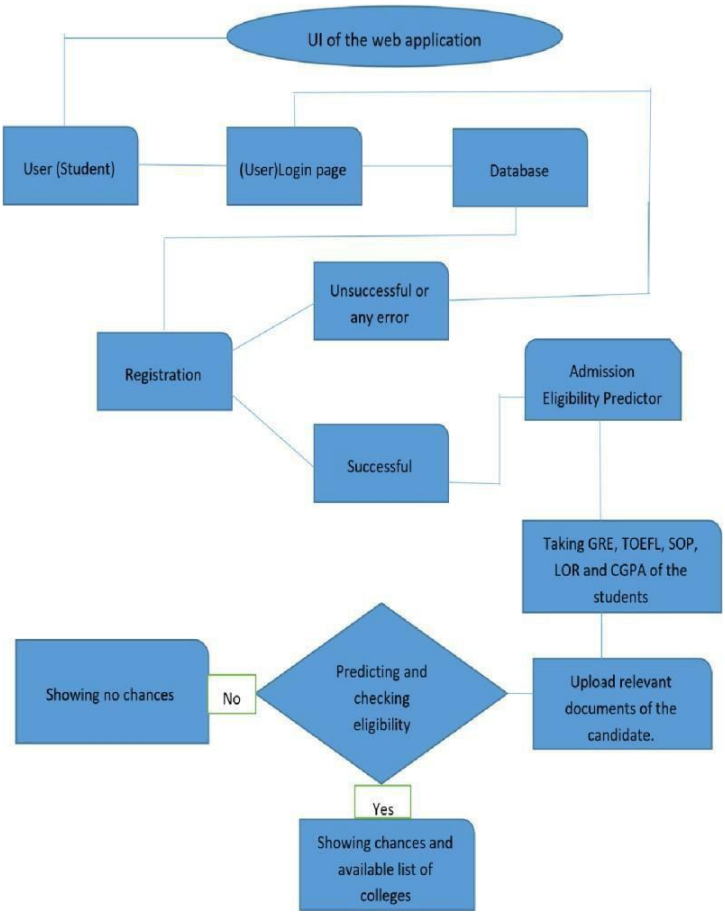
4.2 NON – FUNCTIONAL REQUIREMENTS:

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Filters the universities based on the user profile.
NFR-2	Security	User details are secured from unauthorized parties.
NFR-3	Reliability	The users can find universities based on their preferred location and results.
NFR-4	Performance	The website will provide the list of universities within 30 seconds.
NFR-5	Availability	Students across India can access the website anytime.
NFR-6	Scalability	The solution will be helpful for the students in India to know the details about universities they are eligible.

CHAPTER 5

PROJECT DESIGN

5.1 DATA FLOW DIAGRAM:

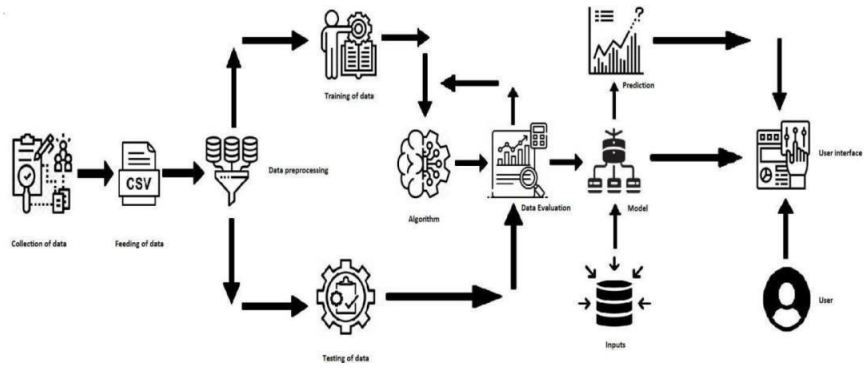


5.2 SOLUTION AND TECHNICAL ARCHITECTURE:

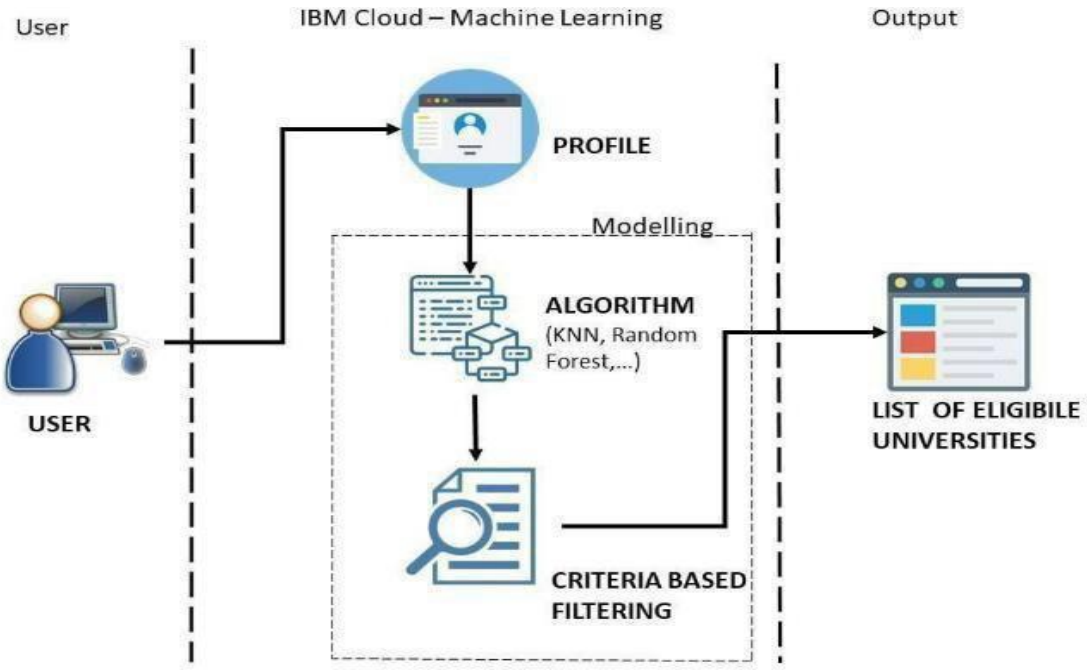
SOLUTION ARCHITECTURE:

Solution architecture is a complex process with many sub processes that bridges the gap between business problems and technology solutions. Its goals are to:

i. Find the best tech solution to solve existing business problems.
ii. Describe the structure, characteristics, behavior and other aspects of the software to project stakeholders.
iii. Define features, development phases and solution requirements.
iv. Provide specifications according to which the solution is defined, managed and delivered.



TECHNICAL ARCHITECTURE:



GUIDELINES:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructure demarcation (Local / Cloud).
3. Indicate interface to machine learning models.
4. Include necessary machine learning algorithms.
5. Indicate Data Storage components / services.
6. Provide the list of all eligible universities along with its description.

5.3 USER STORIES:

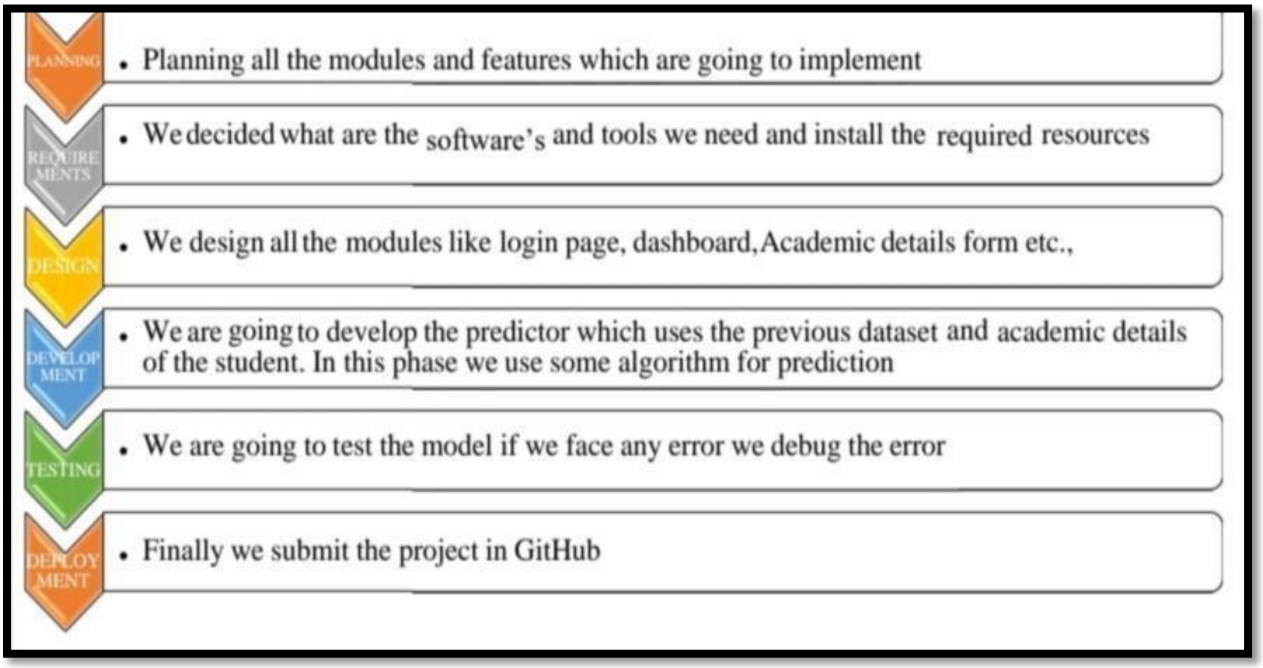
Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, Password and confirming my password.	We can access my account / User dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	We can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Face book	As a user, I can register for the application through Face book	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	We can register and access the dashboard	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	We can access various pages	High	Sprint-1
	Dashboard	USN-6	As a user , I cansearch for various universities	We can access several pages	High	Sprint-1

	Search	USN-7	As a user , I can search for Universities with different field	We can receive information related to universities on various locations	High	Sprint-2
	View	USN-8	As a user , I can view the University details	We will get the information on seat availability, eligibility criteria.	High	Sprint-2
	Receive notification	USN-9	As a user, I will receive notifications about the Suggested universities based on student marks	We will get frequent updates of the preferred universities	Low	Sprint-2
	Chatwith expert	USN-10	As a user, I can chat with the expert for clarifications	We can clear my doubts through chat with expert option	Medium	Sprint-2
Admin	Analysis	USN-11	As an admin, I will analyze the given dataset	We can analyzethe dataset	High	Sprint-2
	Predict	USN-12	As an admin, I will predict the admission	We can predict eligibilityfor admission	High	High

CHAPTER 6

PROJECT PLANNING AND SCHEDULING

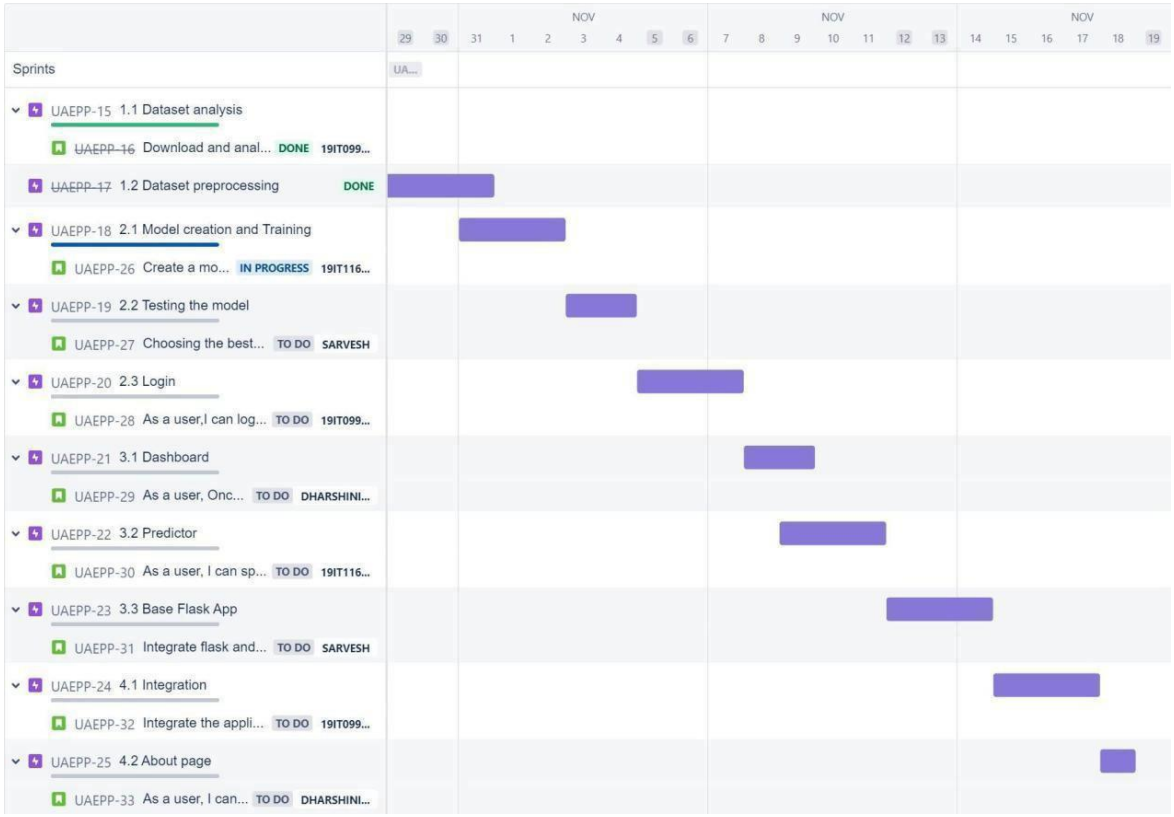
6.1 SPRINT PLANNING AND ESTIMATION:



6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	University Registration	USN-1	As a student, I can register for the application by entering my email by confirming my password.	2	High	2
Sprint -1		USN-2	As a student, I will receive a confirmation email once I have registered for the application from the university.	1	High	1
Sprint-2		USN-3	As a student, I can register for the application through university by uploading my mark statements. Upload original copy of the mark sheets.	2	Low	2
Sprint-3		USN-4	As a student, I can register for the application through Gmail with all eligibility. Students can upload extra course completion certificates.	2	Medium	2
Sprint-4	Login by user name	USN-5	As a student, I can log into the application by entering email and password.	1	High	2
	Dashboard		Check dashboard and upload the details according to university criteria.			4

6.3 REPORTS FROM JIRA:



CHAPTER 7

CODING & SOLUTION:

7.1 FEATURE 1:

Importing the required Libraries

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

Reading the Dataset

```
df = pd.read_csv('Admission_Predict.csv')
df.head ()
```

	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
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

Preparing the Dataset

```
# Dropping the Serial No. column
df.drop("Serial No.", axis=1, inplace=True)
df.head ()
```

	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

Analyzing the Dataset

```
df.head ()
```

	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

```
df.tail ()
```

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance Of Admit
395	324	110	3	3.5	3.5	9.04	1	0.82
396	325	107	3	3.0	3.5	9.11	1	0.84
397	330	116	4	5.0	4.5	9.45	1	0.91
398	312	103	3	3.5	4.0	8.78	0	0.67
399	333	117	4	5.0	4.0	9.66	1	0.95

 # Getting the size of the dataset

```
Print ('No of Rows in the dataset: ',{df.shape[0]})
Print ('No of Columns in the dataset: ',{df.shape[1]})
```

No of Rows in the dataset: {400}
No of Columns in the dataset: {8}

Statistical summary of the data set

```
df.describe ()
```

	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
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

df.skew ()

GRE	-0.062893
TOEFL Score	0.057216
University Rating	0.171260
SOP	-0.275761
LOR	-0.106991
CGPA	-0.065991
Research	-0.191582
Chance of Admit	-0.353448

dtype: float64

 # getting info of the dataset

df.info ()

Range Index: 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

Checking for null values

df.isnull ().sum()

GRE Score	0
TOEFL Score	0
University Rating	0
SOP	0
LOR	0
CGPA	0
Research	0
Chance of Admit	0

dtype: int64

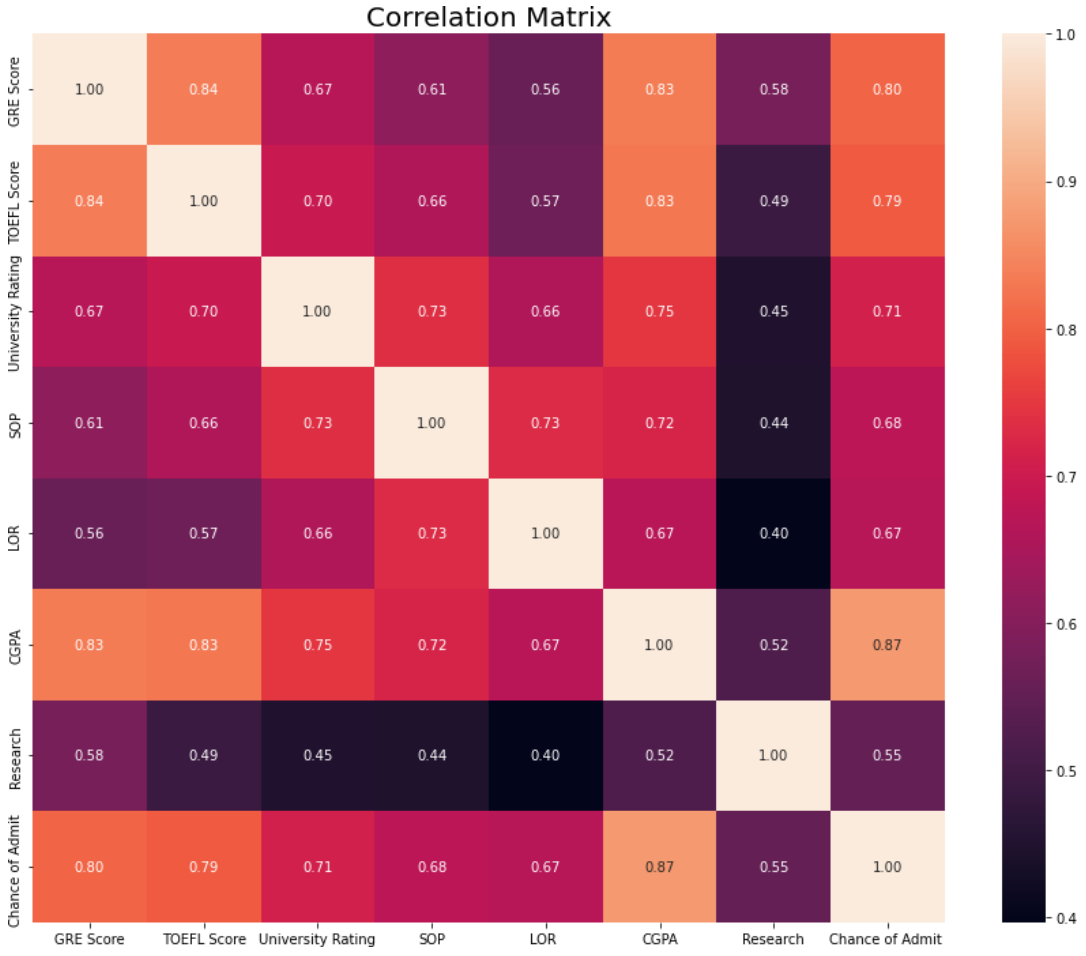
Data Correlation

df.corr ()

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
GRE Score	1.000000	0.835977	0.668976	0.612831	0.557555	0.833060	0.580391	0.802610
TOEFL Score	0.835977	1.000000	0.695590	0.657981	0.567721	0.828417	0.489858	0.791594
University Rating	0.668976	0.695590	1.000000	0.734523	0.660123	0.746479	0.447783	0.711250
SOP	0.612831	0.657981	0.734523	1.000000	0.729593	0.718144	0.444029	0.675732
LOR	0.557555	0.567721	0.660123	0.729593	1.000000	0.670211	0.396859	0.669889
CGPA	0.833060	0.828417	0.746479	0.718144	0.670211	1.000000	0.521654	0.873289
Research	0.580391	0.489858	0.447783	0.444029	0.396859	0.521654	1.000000	0.553202
Chance of Admit	0.802610	0.791594	0.711250	0.675732	0.669889	0.873289	0.553202	1.000000

#plotting the correlation matrix as a heatmap

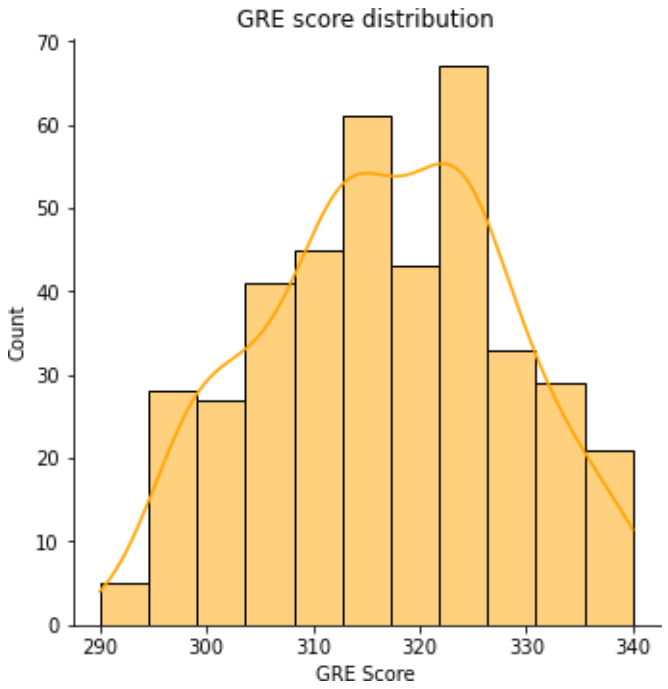
```
corr_matrix = df.corr ()
plt.figure (figsize = (15, 12))
sns.heatmap (corr_matrix, annot=True, fmt='0.2f')
plt.title ("Correlation Matrix", fontsize = 20)
plt.show ()
```



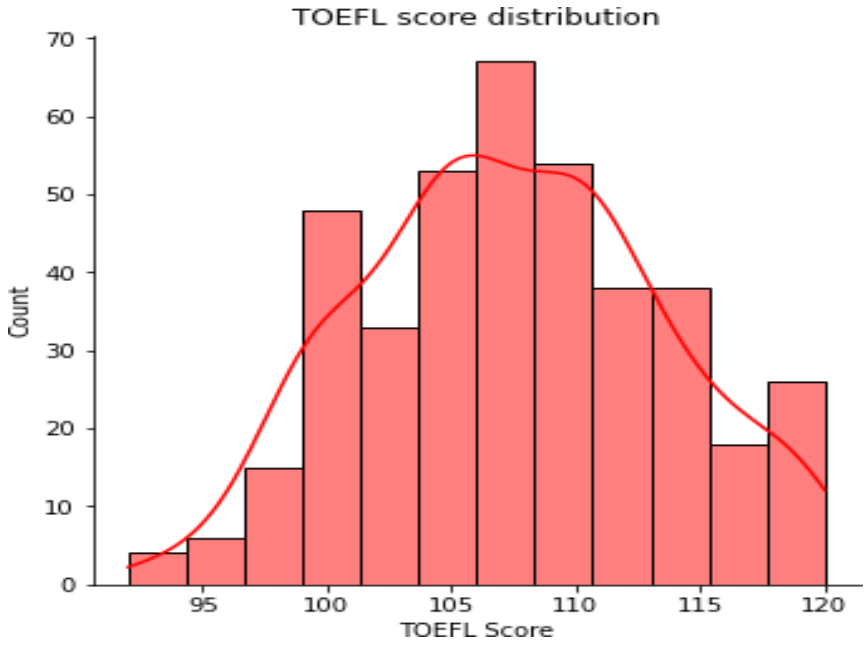
Data Visualization

Univariate Analysis:

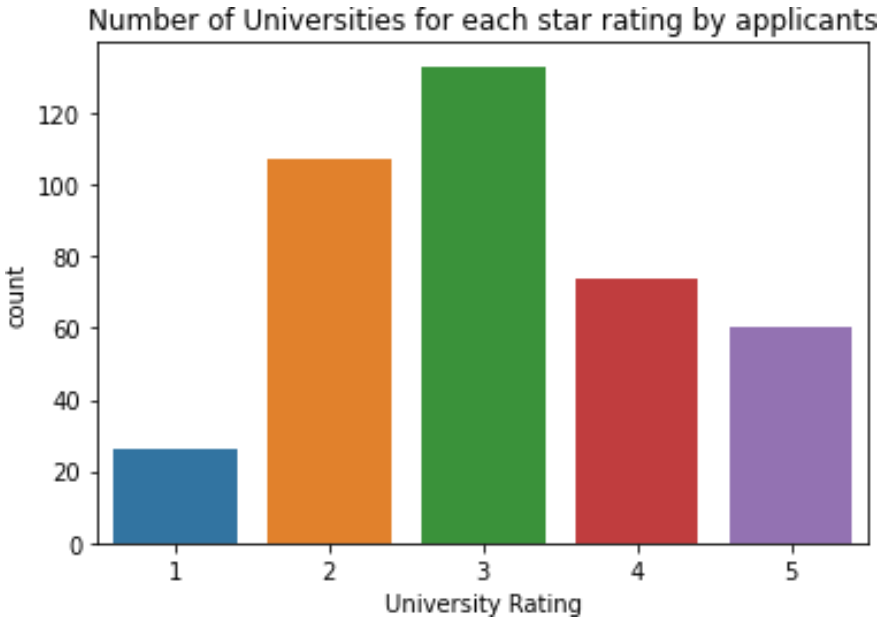
```
sns.displot(x=df["GRE Score"], kde=True, color='orange')
plt.title ("GRE score distribution");
```



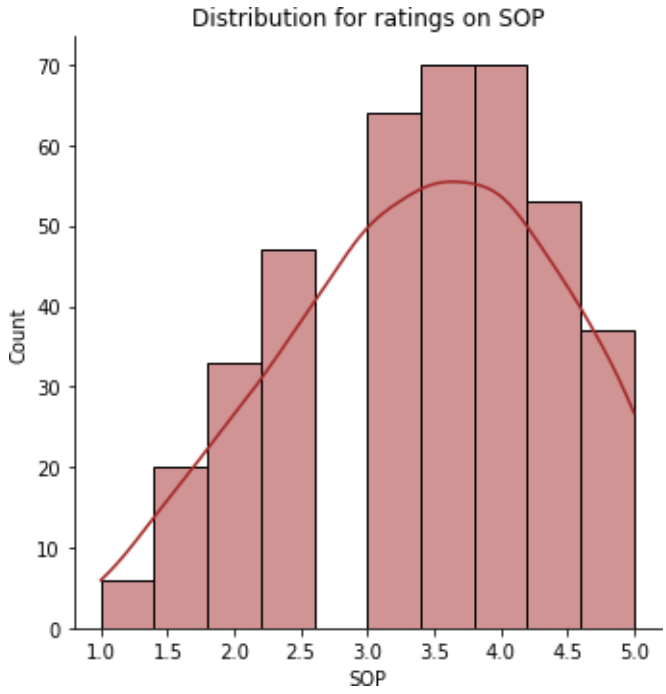

```
sns.displot(x=df["TOEFL Score"], kde=True, color='red')
 plt.title ("TOEFL score distribution");
```



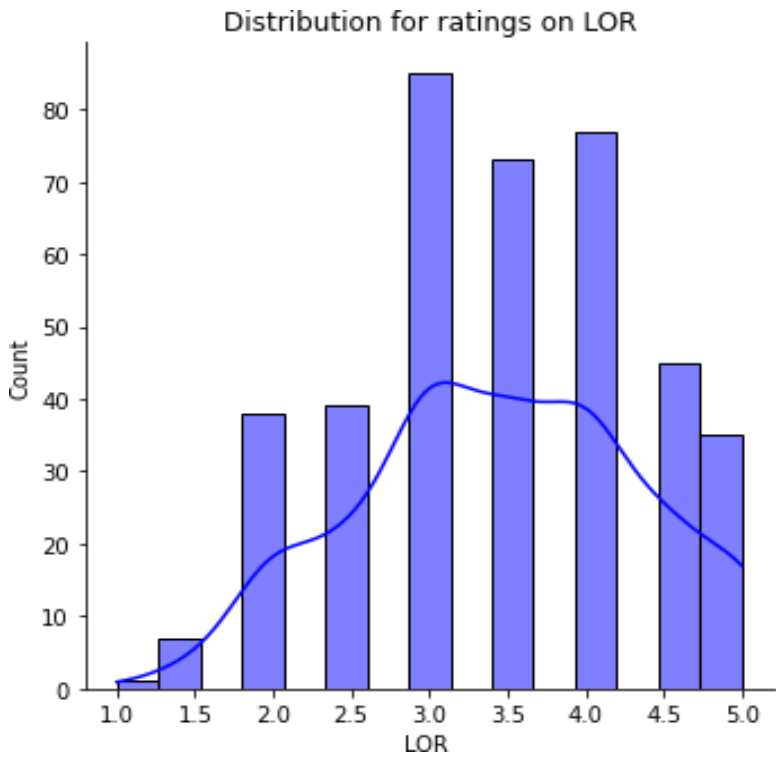
```
sns.countplot(x=df["University    Rating"]);
 plt.title ("Number of Universities for each star rating by applicants)
```



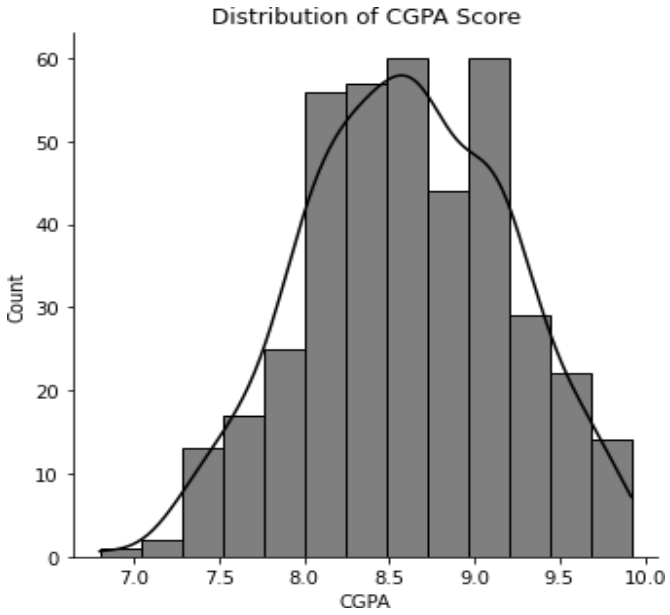

```
sns.displot(x=df["SOP"], kde=True, color='brown');
plt.title("Distribution for ratings on SOP");
```



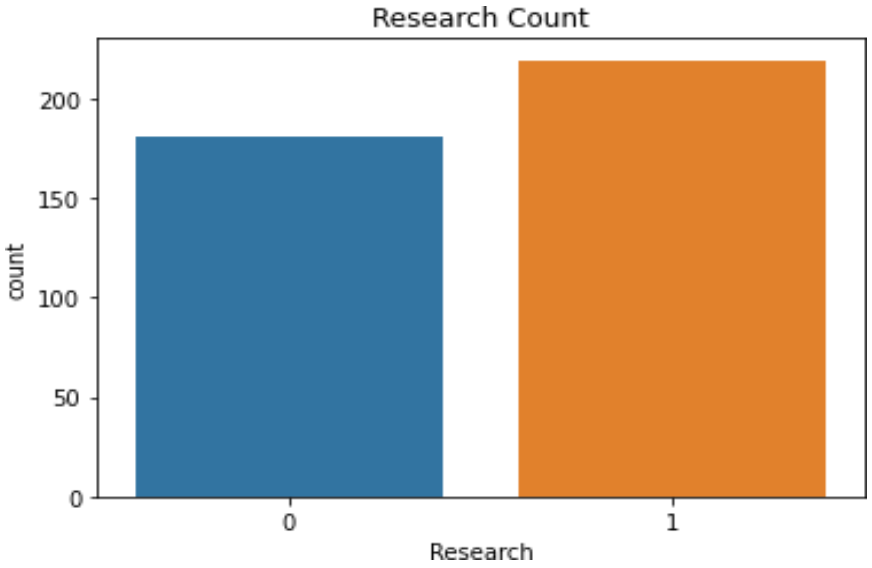
```
sns.displot(x=df["LOR "], kde=True, color='blue');
plt.title ("Distribution for ratings on LOR");
```



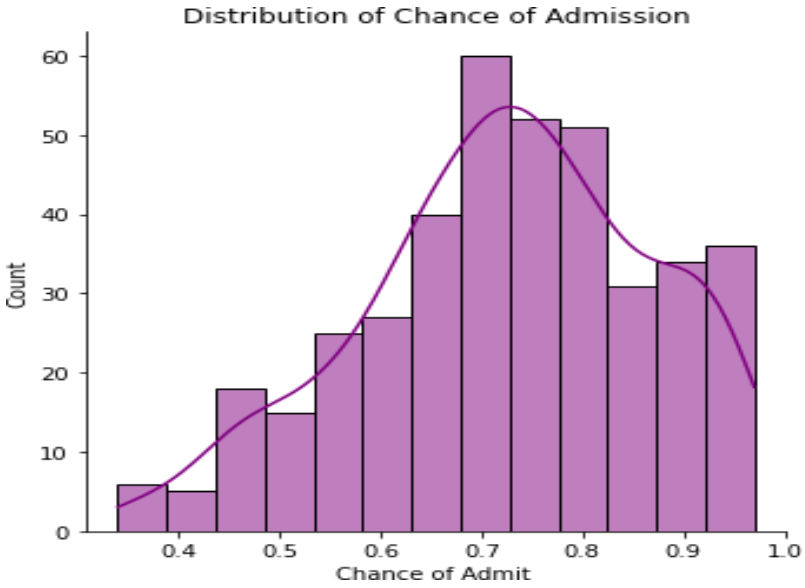

```
sns.displot(x=df ["CGPA"], kde=True, color='black');
 plt.title ("Distribution of CGPA Score");
```



```
sns.countplot(x=df["Research"]);
plt.title ("Research Count");
```




```
sns.displot(x=df["Chance of Admit "], kde=True, color='purple');
 plt.title ("Distribution of Chance of Admission");
```

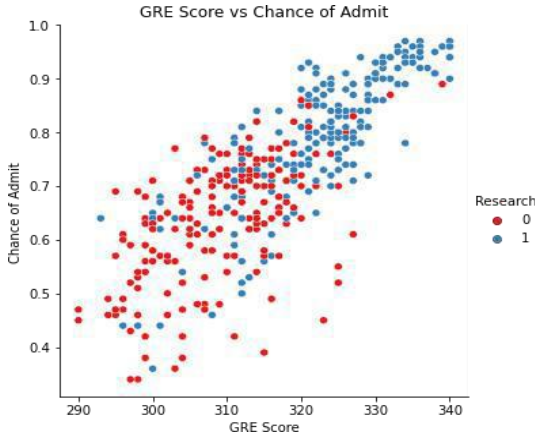


```
df ["GRE Score"].plot (kind = 'hist', bins = 200,figsize = (6,6))
plt.title ("GRE Scores")
plt.xlabel ("GRE Score")
plt.ylabel  ("Frequency")
plt.show ()
```

Bivariate Analysis:

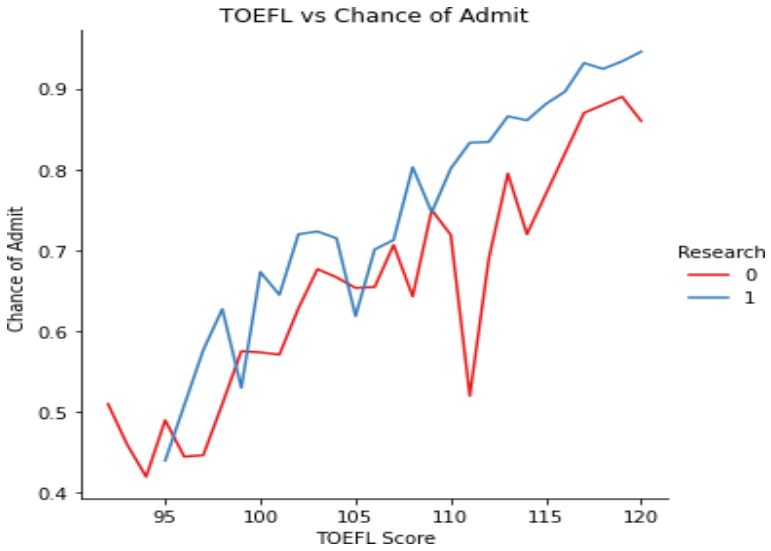
Plotting data which have high correlation:

```
sns.relplot (data=df,x="GRE Score",y="Chance of Admit ",hue="Research", palette="Set1")
plt.title ("GRE Score vs. Chance of Admit")
```

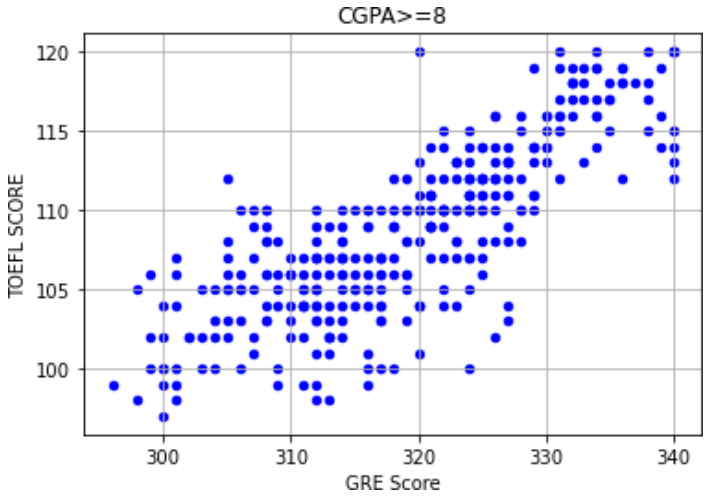


```
plt.show ()
```

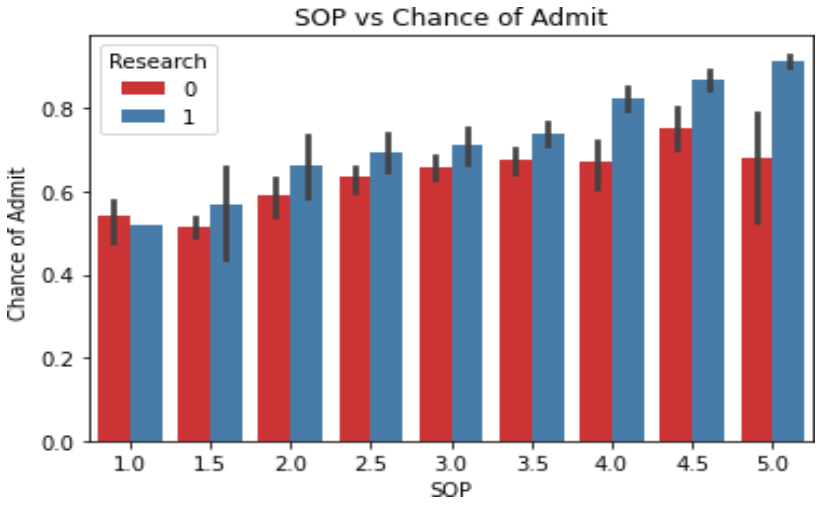
```
 sns.relplot (data=df,x="TOEFL Score",y="Chance of Admit
",hue="Research",kind="line",ci=None, palette="Set1")
plt.title ("TOEFL vs. Chance of Admit")
plt.show ()
```



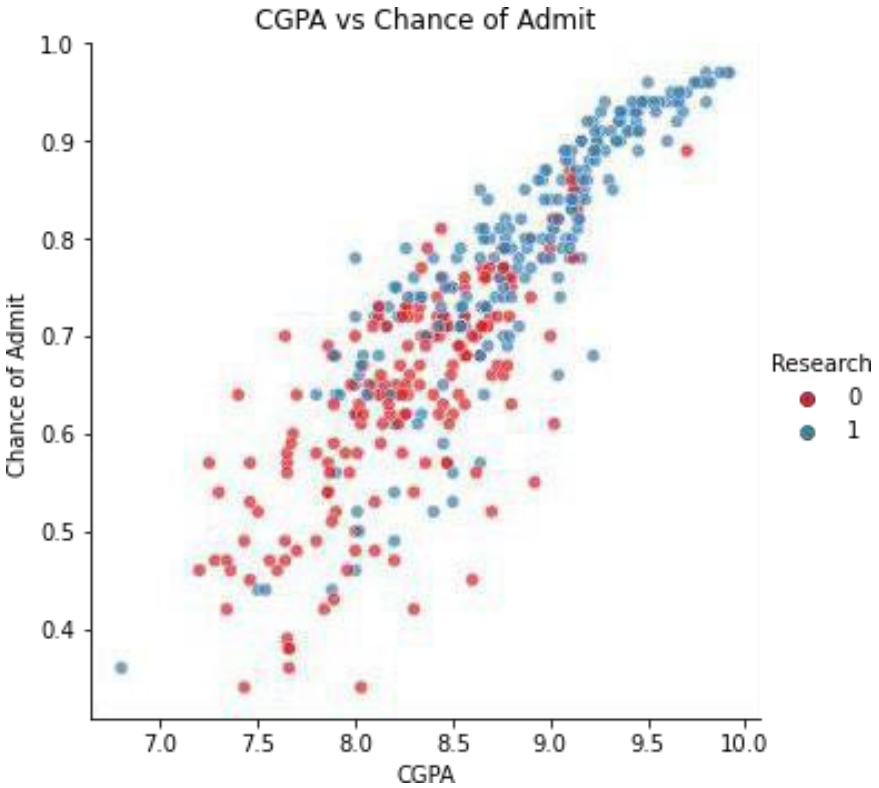

```
df [df.CGPA >= 8].plot(kind='scatter', x='GRE Score', y='TOEFL
Score',color="BLUE")
plt.xlabel ("GRE Score")
plt.ylabel ("TOEFL SCORE")
plt.title ("CGPA>=8")
plt. grid(True)
 plt.show ()
```



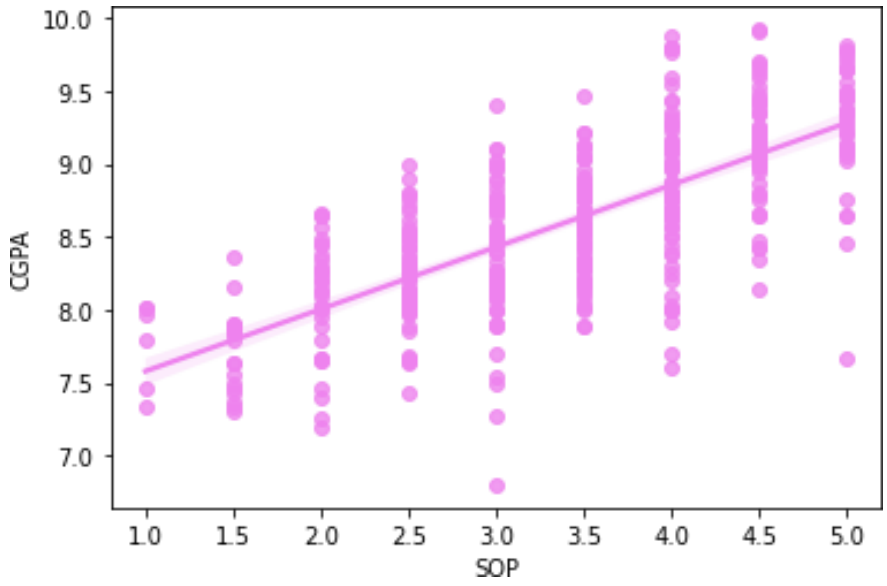
```
sns.barplot (data=df,x="SOP",y="Chance of Admit ",palette="Set1",hue="Research")
plt.title ("SOP vs. Chance of Admit")
plt.show ()
```



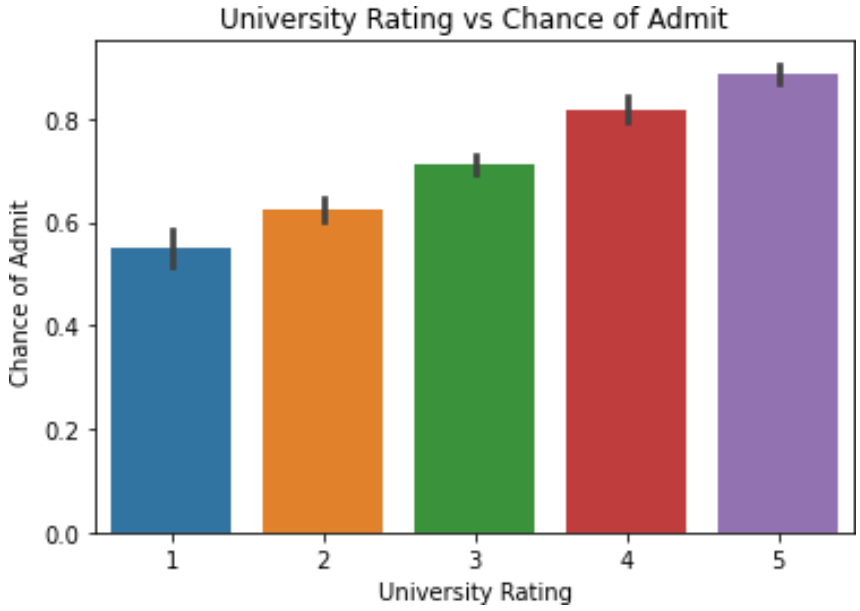

```
sns.barplot (data=df,x="LOR ",y="Chance of Admit ", palette="Set1",hue="Research")
plt.title ("LOR vs. Chance of Admit")
plt.show ()
```



```
sns.regplot (df ['SOP'], df ['CGPA'], color='violet')
```



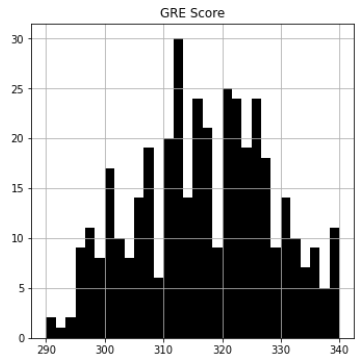

```
sns.barplot (data=df, x="University Rating",y="Chance of Admit ")
plt.title ("University Rating vs. Chance of Admit")
plt.show ()
```

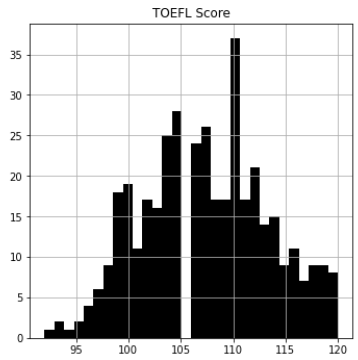


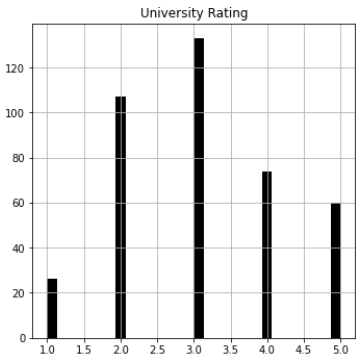
df.hist (bins = 30, figsize = (20, 20), color = 'black')

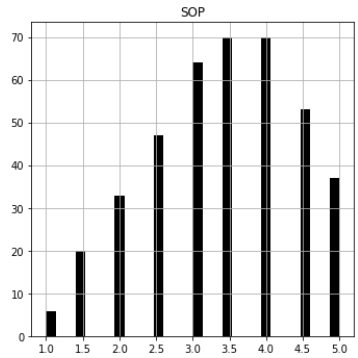
array ([[,

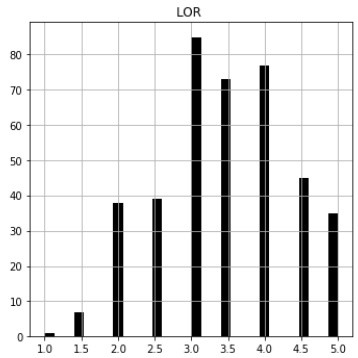
],
[,
,
],
[,
,
]], dtype=object)

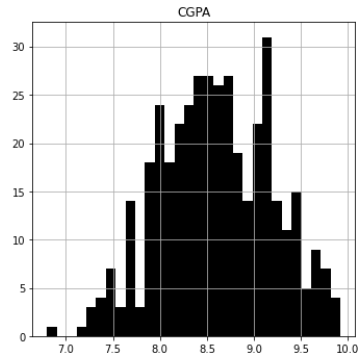


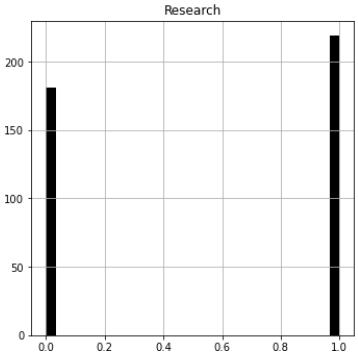


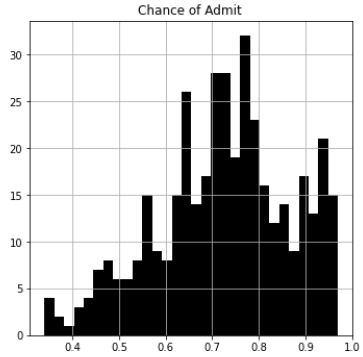








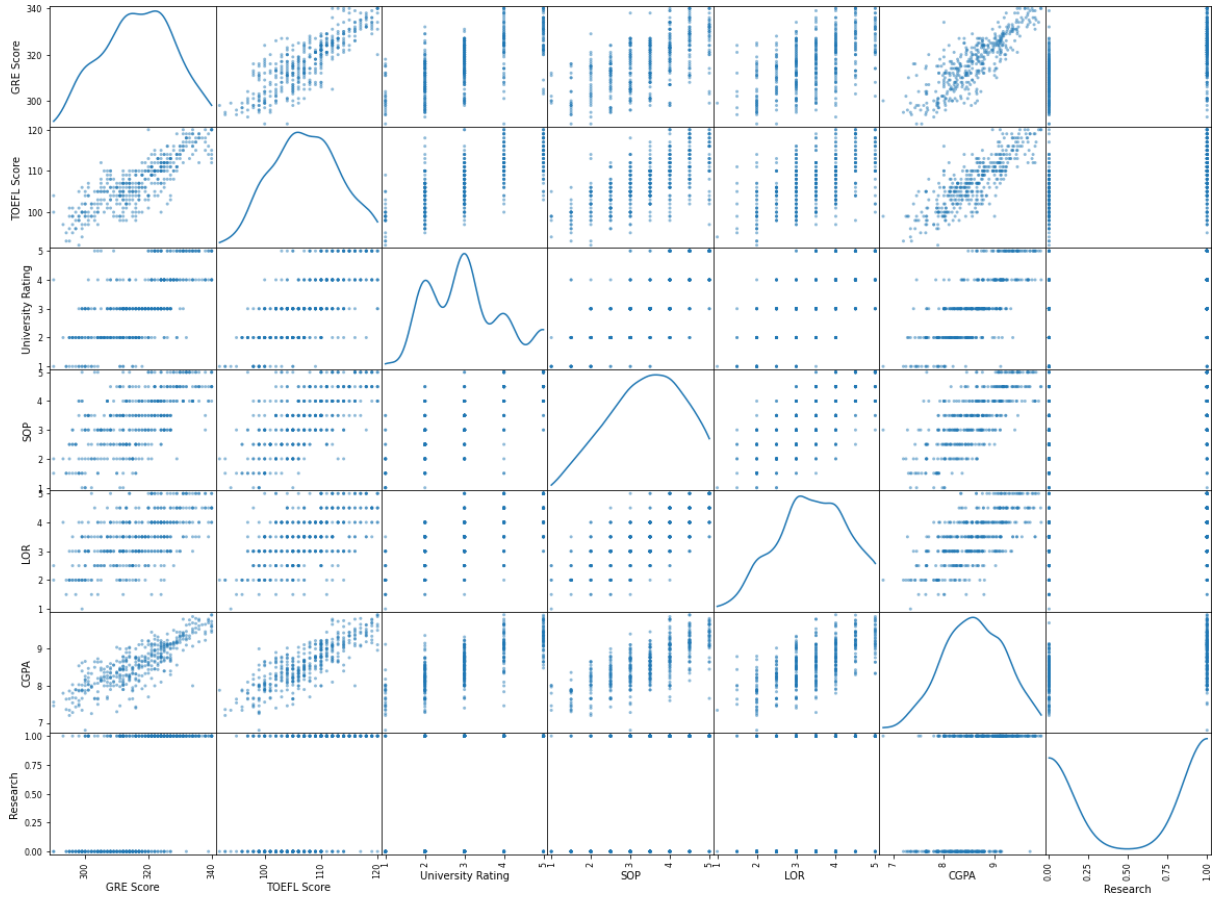




Multivariate Analysis:

sns.pairplot (df, hue='Research', palette='Set1')

```
pd.plotting.scatter_matrix (df.loc [:,"GRE Score":"Research"],diagonal="kde",figsize=(20,15))
plt.show
```



Importing the required libraries for regression analysis

```
import importlib.util from sklearn
import metrics from sklearn
import datasets  from sklearn
import linear_modelfrom sklearn.model_selection
import  train_test_split  from  sklearn.linear_model
import LogisticRegression from sklearn.ensemble
import GradientBoostingRegresso from sklearn. Preprocessing
import scale from sklearn. Preprocessing
import MinMaxScaler from sklearn. Metrics
import accuracy_score,confusion_matrix from sklearn. Metrics
import recall score, roc_auc_score from sklearn. Metrics
import mean_squared_error, r2_score,    mean_absolute_error
```

Splitting Dependent and Independent Columns

```
X=df.drop (['Chance of Admit '], axis=1)
 y=df ['Chance of Admit ']
```

```
X.head ()
```

	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

```
X.shape
```

```
(400, 7)
```

```
y.head ()
```

```
0    0.92
1    0.76
2    0.72
3    0.80
4    0.65
Name: Chance of Admit, dtype: float64
y.shape
```

```
(400 ,)
```

Splitting the data into Train and Test

X_train, X_test, y_train, y_test = train_test_split(X, y, test size=0.2, random_state=1)

Scaling the data

Scale = MinMaxScaler ()

X_train [X_train.columns] = scale.fit_transform (X_train [X_train.columns])
X_test [X_test.columns] = scale.transform (X_test [X_test.columns])

X_train.head ()

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
93	0.22	0.178571	0.25	0.50	0.428571	0.250000	1.0
23	0.88	0.964286	1.00	1.00	0.857143	0.919118	1.0
299	0.30	0.714286	0.50	0.50	0.571429	0.533088	0.0
13	0.34	0.607143	0.50	0.75	0.428571	0.294118	1.0
90	0.56	0.500000	0.25	0.75	0.714286	0.264706	1.0

X_test. Head ()

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research
398	0.44	0.392857	0.50	0.625	0.714286	0.580882	0.0
125	0.20	0.285714	0.50	0.250	0.428571	0.536765	1.0
328	0.68	0.714286	0.75	0.750	0.571429	0.577206	1.0
339	0.68	0.535714	1.00	0.625	0.714286	0.536765	1.0
172	0.64	0.642857	0.75	0.750	0.709559	0.709559	1.0

Model Building

Gradient Boosting Regression

```
rgr = GradientBoostingRegressor ()
rgr.fit (X_train, y_train)

GradientBoostingRegressor  ()


rgr. Score (X_test, y_test)

0.7845693721713658

y_predict=rgr. Predict (X_test)

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: 0.046120348671317354
Mean Squared Error: 0.004982358380692498
Root Mean Squared Error: 0.07058582280240486
```

Logistic Regression

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


from sklearn.linear_model._logistic import LogisticRegression
lore = LogisticRegression(random state=0, max_iter=1000)
lr = lore.fit(X_train, y_train)
y_pred = lr.predict(X_test)
```

Model Evaluation

```
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: 0.9125
Recall Score: 1.0
ROC AUC Score: 0.6111111111111112
Confusions Matrix:
 [[ 2  7]
 [ 0 71]]
```

Save the Model

```
import pickle
pickle.dump(lr, open("university.pkl", 'wb'))
model = pickle. load (open ("university.pkl", 'rb'))
```

7.2 FEATURE 2:

```
<!DOCTYPE html>
<html>
<head>
  <SCRIPT language=Javascript>


  </SCRIPT>
<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/
font-awesome/4.7.0/css/font-awesome.min.css">
<style>
body {
  font-family: Arial, Helvetica, sans-serif;
  color: brown;
}

* {
  box-sizing: border-box;
}

/* style the container */
.container {
  position: relative;
  border-radius: 5px;
  /* background-color:aqua; */
  background-image: url(https://img.freepik.com/free-vector/
  group-college-university-students-hanging-out_74855-5251.
  jpg?w=1060&t=st=1668931627~exp=1668932227~hmac=dca75d9daca94bce5f9de8b6c988159453fe5eea
```

```
  fd45aaf087e4167ceeb11b52);
  background-size:cover;
  background-position: center;
  background-repeat: no-repeat;
  height: 100vh;
  padding: 20px 0 30px 0;
}

/* style inputs and link buttons */
input,
.btn {
  width: 100%;
  padding: 12px;
  border: none;
  border-radius: 4px;
  margin: 5px 0;
  opacity: 0.85;
  display: inline-block;
  font-size: 17px;
  line-height: 20px;
  text-decoration: none; /* remove underline from anchors */
}

input:hover,
.btn:hover {
  opacity: 1;
}

li{
```

```
  font-family: Impact, Haettenschweiler, 'Arial Narrow Bold', sans-serif;
  color: black;
}


/* style the submit button */
input[type=submit] {
  background-color: #000000;
  color: white;
  cursor: pointer;
}

input[type=submit]:hover {
  background-color: #151615;
}

/* Two-column layout */
.col {
  float: left;
  width: 50%;
  margin: auto;
  padding: 0 50px;
  margin-top: 6px;
}

/* Clear floats after the columns */
.row:after {
  content: "";
  display: table;
```

```
  clear: both;
}

/* vertical line */
.vl {
  position: absolute;
  left: 50%;
  transform: translate(-50%);
  border: 2px solid rgb(8, 8, 8);
  height: 490px;
}


/* hide some text on medium and large screens */
.hide-md-lg {
  display: none;
}

/* bottom container */
.bottom-container {
  text-align: center;
  background-color: aliceblue;
  border-radius: 0px 0px 4px 4px;
}

/* Responsive layout - when the screen is less than 650px wide, make the two
columns stack on top of each other instead of next to each other */
@media screen and (max-width: 650px) {
  .col {
```

```
    width: 100%;
    margin-top: 0;
  }
  /* hide the vertical line */
  .vl {
    display: none;
  }
  /* show the hidden text on small screens */
  .hide-md-lg {
    display: block;
    text-align: center;
  }
}
</style>
</head>
<body>

<div class="container">
  <form action="{{url_for('predict')}}" method="post">
    <div class="row">
      <marquee class="bottom-container"><h2>If studying abroad is your dream, making
        it simple is ours!</h2></marquee>
      <div class="vl">
        <span class="vl-innertext"></span>
      </div>

      <div class="col">
      <p><b>A simple Web App to predict the chances of getting an admit based
        on Student's profile</p></b>
```

```
      <h3>Input Guide</h3>
      <ul>
            <li>GRE Score (out of 340)</li>
            <li>TOEFL Score (out of 120)</li>
            <li>University Rating (out of 5) - the category of the target university</li>
            <li>Statment of Purpose {SOP} Strength (out of 5)</li>
            <li>Letter of Recommendation {LOP} Strength (out of 5)</li>
            <li>Undergraduate CGPA (out of 10)</li>
            <li>Research Experience (0 for NONE and 1 for YES)</li>

      </ul>
      </div>

      <div class="col">
        <div class="hide-md-lg">
        </div>

            <input type="number" name="GRE Score" placeholder="GRE Score" required="required"
            min="0" max="340"/>
            <input type="number" name="TOEFL Score" placeholder="TOEFL Score" required="required"
            min="0" max="120"/>
            <input type="number" name="University Rating" placeholder="University Rating" required=
            "required" min="1" max="5"/>
            <input type="number" name="SOP" placeholder="SOP" required="required" onkeypress="
            return check(event,value)" step="0.1" min="1" max="5"/>
            <input type="number" name="LOR" placeholder="LOR" required="required" onkeypress="
            return check(event,value)" step="0.1" min="1" max="5"/>
            <input type="number" name="CGPA" placeholder="CGPA" required="required" onkeypress="
            return check(event,value)" step="0.01" min="1" max="10"/>
```

```
            <input type="number" name="Research" placeholder="Research" required="required" min="0"
            max="1"/>

            <input type="submit" value="Predict"></input>

            <h4 style="text-align: center;"></h4>
      </div>

    </div>
  </form>

</div>


</body>
</html>
```

CHAPTER 8

TESTING

8.1 TEST CASES:

TEST CASE ID	FEATURE TYPE	COMPONENT	TEST SCENARIO	PRE REQUISITES	STEPS TO EXECUTE	TEST DATA	EXPECTED RESULT	ACTUAL RESULT	STATUS	COMMENT	TC (Y/ N)	BUG ID	EXECUTED BY
Login Page_ TC_ 001	UI	Login Page	Tested with all the UI component and input fields in thehome page	HTML, CSS	1. Needto give URL first. 2. Check with all the elements in the UIthat it is displayed or not.		All the UI Components and the input fieldshaveto be function properly	Working As expecte d	Pass		N		AARTHI.M
Login Page_ TC_ 002	Functional	Login Page	Verifying whether the userhas been authenticated to enter into our predictor page.	HTML, CSS	1. Need to click on the predictor tab in the navigation bar. 2. Check with all the elements in the UI that it is displayedor not.		All the UI Component has to be function properly and users has to navigateto the predictor page properly.	Working as expected	Pass		N		GAYATHRI.E
Predict Page_T C _003	UI	Predict Page	Verify whether all the UI elements in the Prediction page are functioning properlyor not.	HTML, CSS, Flask	Check with all the elementsin the UI that it is displayed or not.		All the UI components has to be function properly.	Working as expected	Pass		N		SHAMRITHA. P.R
Predict Page_T C_ 004	Functional	Predict Page	Do enter the values in the input fields and click on predict.	Flask	1. Need to enter 7 valuesfor each attribute and click on predict.	200 100 4 4 4 9 No Research	Navigateto the prediction page and predict the accurate result.	Working as expected	Pass		N		SHAFIYA AFSAN.Z
Output Page_T C_ 005	Functional	Chance Page	Verify whether it is navigates to chancepage only if the appropriate values are entered.	Flask	2. Needto enter 7 valuesfor each attribute and click on predict. 3. If prediction equals one, a chance page is displayed	Prediction = You have a chance.	Do navigate to chance page	Working as expected	Pass		N		AARTHI.M

Output Page _TC _006	Functional	No chance Page	Checking whether it is redirected to no chance page or not.		1. Need to enter the URL. 2. Need to enter thevalues and click on predict. 3. It has to be redirected to no chancepage if the predictor result is zero.	Prediction = You don't have a chance.	Do redirect to no chance page.	Working as expected	Pass		N		SHAFIYA AFSAN.Z

S.NO	TEST SCENARIOS
1	Tested with all the UI components and input fields in the home page.
2	Verifying whether the user has been authenticated to enter into our predictor page.
3	Verify whether all the UI elements in the Prediction page are functioning properly or not.
4	Do enter the values in the input fields and click on predict.
5	Verify whether it is navigates to chance page only if the appropriate values are entered.
6	Checking whether it is redirected to no chance page or not.

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 [Product Name] 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	SEVERITY 5
BY DESIGN	11	4	2	3	**20**
DUPLICATE	1	0	3	0	**4**
EXTERNAL	3	3	0	1	**7**
FIXED	10	2	4	20	**36**
NOT REPRODUCED	0	0	1	0	**1**
SKIPPED	0	0	1	1	**2**
WON'T FIX	0	5	2	1	**8**
TOTAL	**25**	**14**	**13**	**26**	**78**

3. TEST CASE ANALYSIS:

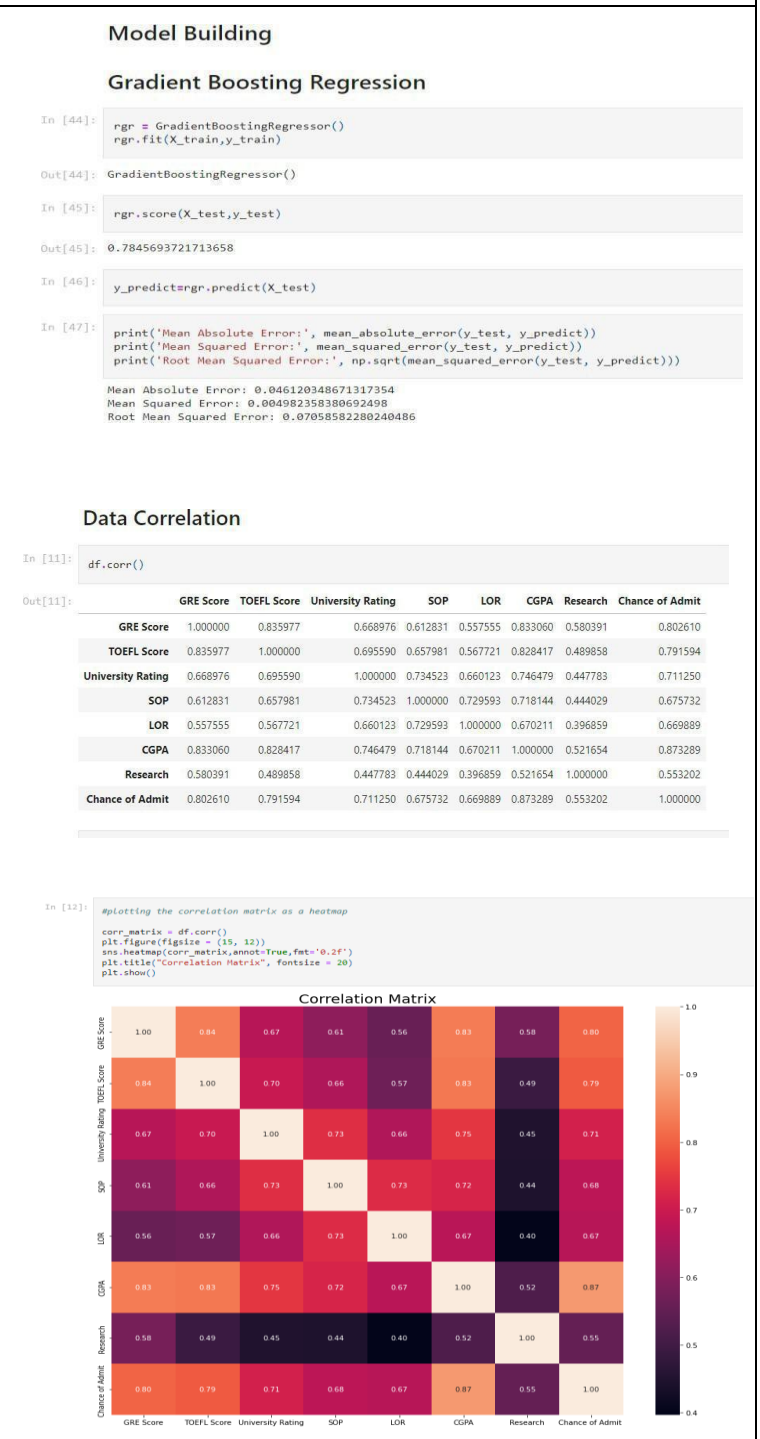
This report shows the number of test cases that have passed, failed, and untested.

SECTION	TOTAL CASES	NOT TESTED	FAIL	PASS
LOGIN PAGE	4	0	0	4
INDEX PAGE	8	0	0	8
PREDICT PAGE	2	0	0	2

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS:

S.NO	PARAMETER	VALUES	SCREENSHOT
1.	**Metrics**	**Regression Model:** MAE-, MSE-, RMSE R2 Score- **Classification Model:** Confusion Matrix-, Accuracy Score- and Classification Report-	

Model Building

Gradient Boosting Regression

```
In [44]: rgr = GradientBoostingRegressor()
         rgr.fit(X_train,y_train)

Out[44]: GradientBoostingRegressor()

In [45]: rgr.score(X_test,y_test)

Out[45]: 0.7845693721713658

In [46]: y_predict=rgr.predict(X_test)

In [47]: 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: 0.046120348671317354
Mean Squared Error: 0.004982358380692498
Root Mean Squared Error: 0.07058582280240486
```

Data Correlation

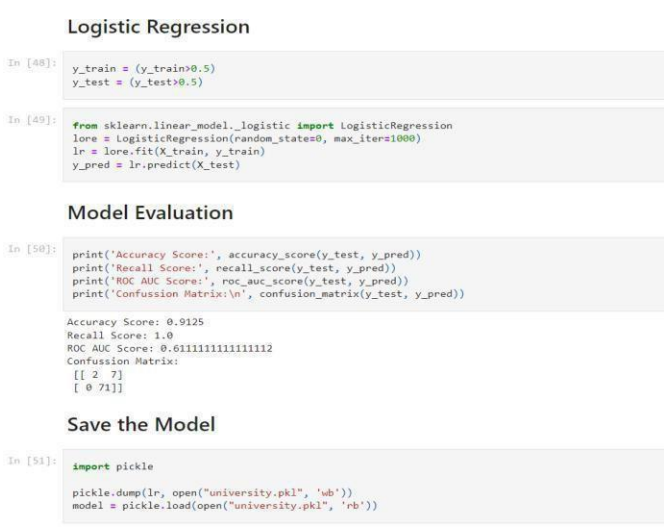
```
In [11]: df.corr()
```

Out[11]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
GRE Score	1.000000	0.835977	0.668976	0.612831	0.557555	0.833060	0.580391	0.802610
TOEFL Score	0.835977	1.000000	0.695590	0.657981	0.567721	0.828417	0.489858	0.791594
University Rating	0.668976	0.695590	1.000000	0.734523	0.660123	0.746479	0.447783	0.711250
SOP	0.612831	0.657981	0.734523	1.000000	0.729593	0.718144	0.444029	0.675732
LOR	0.557555	0.567721	0.660123	0.729593	1.000000	0.670211	0.396859	0.669889
CGPA	0.833060	0.828417	0.746479	0.718144	0.670211	1.000000	0.521654	0.873289
Research	0.580391	0.489858	0.447783	0.444029	0.396859	0.521654	1.000000	0.553202
Chance of Admit	0.802610	0.791594	0.711250	0.675732	0.669889	0.873289	0.553202	1.000000

```
In [12]: #plotting the correlation matrix as a heatmap

         corr_matrix = df.corr()
         plt.figure(figsize = (15, 12))
         sns.heatmap(corr_matrix,annot=True,fmt='0.2f')
         plt.title("Correlation Matrix", fontsize = 20)
         plt.show()
```

2.	**Tune the model**	Hyper parameter Tuning - Validation Method -	

CHAPTER 10

ADVANTAGES & DISADVANTAGES:

ADVANTAGES:

- It gives an overall accuracy of 94%, which is really high.
- The dataset consists of all possible attributes needed for prediction.
- Confidence booster if results are positive.
- Students can also change their scores to see how they affect the overall prediction results and focus more on that area.

DISADVANTAGES:

- The model is built in such a way that the prediction is positive only if the chance of admit percent is greater than 80%. Even if the result is 79%, the prediction would be NO.
- The complexity of the examinations is not considered. Therefore, the results may vary every year for the same set of attribute
- The dataset used for training the model is of comparatively small size. Therefore, the model cannot be relied on to take accurate real-time decisions.

CHAPTER 11

CONCLUSION:

The dataset is trained with different ML model. The ML models used to train our dataset are KNN , Logistic Regression , Random Forest , SVM each having accuracy of 86%,88%,93%,89% respectively. Logistic Regression algorithm is finally selected to be used in our model. The Machine Learning model is integrated using flask for our web application. At long last, understudies can have an open –source AI model which will assist the understudies with knowing their opportunity of entrance into a specific college with high exactness.

CHAPTER 12

FUTURE SCOPE:

A real-time project can be developed by gathering data from institutions. The data can be processed and trained using big data frameworks like spark and MLLib can be used to train the data using different machine learning models.

CHAPTER 13

APPENDIX:

SOURCE CODE:

APP.PY

```python
import pandas as pd
from flask import Flask, request, jsonify, render_template,redirect,url_for
import pickle

app = Flask(__name__,template_folder='Template')
model = pickle.load(open(r'C:\ProgramData\UAEP\Application Building\university.pkl', 'rb'))



@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', methods=['GET','post'])
def predict():

    GRE_Score = int(request.form['GRE Score'])
    TOEFL_Score = int(request.form['TOEFL Score'])
    University_Rating = int(request.form['University Rating'])
    SOP = float(request.form['SOP'])
    LOR = float(request.form['LOR'])
    CGPA = float(request.form['CGPA'])
    Research = int(request.form['Research'])

    final_features = pd.DataFrame([[GRE_Score, TOEFL_Score, University_Rating, SOP, LOR, CGPA,Research]])
    predict=model.predict(final_features)

    output=predict[0]
    if output > 0.5:
        return redirect(url_for('chance', percent=output*100))
    else:
        return redirect(url_for('no_chance', percent=output*100))



@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])

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

GitHub & Project Demo Link:

Git Hub link: https://github.com/IBM-EPBL/IBM-Project-44582-1660725370

Demo Video link: https://drive.google.com/file/d/1BgT_xnkGqEbVyPjFpcSMHytd04LA-JAQ/view?usp=sharing