[Turnitin]CoreProject_Report_FLY-EASY

by Gelle Abhiram

Submission date: 08-Dec-2021 05:52PM (UTC+0530)
Submission ID: 1719449613
File name: CoreProject_Report_FLY-EASY.pdf (16.48M)
Word count: 3349
Character count: 17483

A CORE PROJECT REPORT ON

FLY EASY

By

Names of the students	**Registration No.**
Gelle Abhiram	1800217C203
Gurram Raghavendra Dinesh	1800219C203
Amarthaluru Paavan Dileep	1800197C203

Supervisor

Dr. Pradeep Kumar Arya

Assistant Professor

Prepared in the partial fulfillment of the

B-tech 4th year Requirements

For the Degree of

Bachelor of Technology

In

Computer Science Department

AT

BML MUNJAL UNIVERSITY

December 2021

Table of Contents

Acknowledgement

The success and outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this along with the completion of my project. All that we have done is only due to such supervision and assistance and we would not forget to thank him.

We respect and thank **Dr. Pradeep Kumar Arya** for providing us with an opportunity to do the project work under his supervision and giving us support and guidance which made us complete the project duly. We are extremely thankful to him for providing such nice support and guidance, although they have a busy schedule.

We perceive this opportunity as a big milestone in our career development. We will strive to use gained skills and knowledge in the best possible way, and we will continue to work on our improvement, to attain desired career objectives.

Thanking You,

Gelle Abhiram (1800217C203)
Gurram Raghavendra Dinesh (1800219C203)
Amarthaluru Paavan Dileep (1800197C203)

BML Munjal University, Gurgaon, Haryana

CANDIDATE'S DECLARATION

We,**Gelle Abhiram, Gurram Raghavendra Dinesh, Amarthaluru Paavan Dileep** hereby declare that the work done in our core project entitled *"FLY EASY"* in fulfillment of completion of 7th semester of Bachelor of Technology (B.Tech) program in the Department of Computer Science and Engineering, BML Munjal University is an authentic record of our original work carried out under the guidance of **Dr. Pradeep Kumar Arya,** Assistant Professor

Due acknowledgements have been made in the text of the project to all other materials used.

This core project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Gelle Abhiram (1800217C203)
Gurram Raghavendra Dinesh (1800219C203)
Amarthaluru Paavan Dileep (1800197C203)

Date: 04/12/2021

1.Problem Statement:

For an aspiring student who wants to apply for higher studies in other countries, the university selection process is a challenging task. Lot of different criteria need to be considered during the application process based on an individual's requirement. Some of them succeed and get admission into their desired programs in desired universities, but some are not because of the academic level of colleges, which they have applied to. This problem can be addressed by modeling a recommender system based on various classification algorithms. In this project based on the Graduate and Undergraduate student dataset and user profile, a list of 10 best universities will be suggested such that it maximizes the chances of a student getting admission into those universities.

2.Objective:

In this report, we discussed the application "FLY EASY," which is a web application for undergraduate and graduate admission seekers that is embedded with a machine learning and web scraping recommender system that can assist students in selecting the best graduate university that matches their academic scores. From the data of successful students who have already gotten admitted into overseas colleges and universities, we have used various data mining techniques to transform a database of students of important information into a universal database format. Following that, we created a machine learning system that uses weighted scores to calculate the similarity of training and test data. We calculated top N similar users for the test users using the K-nearest Neighbor method and a feature-weighted approach, and we recommended Top K universities to users from the N similar users.

3.Introduction:

Many people who desire to pursue higher education apply to various colleges using their academic credentials and standardized test scores such as the SAT, GRE, TOEFL, and IELTS. Students with adequate academic profiles and standardized test scores are accepted into institutions. However, the most important and time-consuming phase in the graduate school application process is university selection. Some of them are accepted into their desired programmes at preferred institutions, while others are not due to the intellectual quality of the colleges to which they have applied. We created the FLY EASY online application to address the problem of students being denied admission due to a lack of applications, while having strong academic credentials. In this study, data mining techniques are utilized to forecast colleges based on information gathered from a database of successful applicants. This data will be modeled into machine learning algorithms to predict the universities and their acceptance rate for the given user academic details.

4.Project Name & Logo:

Project Name: **FLY EASY**

Project Logo:

Fig 4: *Get Graduate – Application UI*

5.Project Methodology:

5.1.Problem:

As discussed above in the problem statement, for an aspiring student who wants to apply for higher studies in other countries, the university selection process is a challenging task. Lot of different criteria need to be considered during the application process based on an individual's requirement.

5.2.Dataset:

Dataset identification will be the first step for creating recommendations involving applications/sites. This data must be arranged with proper labeling so as to develop the recommender system's classification model. Data related for the application procedure is not broadly available for direct consumption from any online source like the internet. However, the entire strategy is dependent on making the most of the information provided. Graduate student information was obtained from www.thegradcafe.com, while undergraduate university student information was obtained from the - https://collegescorecard.ed.gov/data/.

Graduate Student Dataset:

For Graduate Student data, we scraped www.thegradcafe.com website. About 271807 rows of data where information related to raw students was obtained as an outcome of web scraping. Each record will be similar to that of a student's profile. We have got 1949 html pages of the data and need to change it into CSV files.

```
: import matplotlib.pyplot as plt
  import requests
  import urllib.request
  from IPython.core.debugger import Tracer

  url_form = "http://thegradcafe.com/survey/index.php?q=u%2A&t=a&pp=250&o=d&p={0}"
  DATA_DIR = './WebScraped_data/html/'

  if __name__ == '__main__':
      for i in range(1691, 1948):
          url = url_form.format(i)
          handle = urllib.request.urlopen(url)
          html = handle.read()
          html = html.decode('utf8')
          #r = requests.get(url)
          fname = "{data_dir}/{page}.html".format(data_dir=DATA_DIR,page=str(i))
          with open(fname, 'wb') as f:
              f.write(html.encode('UTF-8'))
          print("getting {0}...".format(i))
  getting 1775...
  getting 1776...
  getting 1777...
  getting 1778...
  getting 1779...
  getting 1780...
  getting 1781...
```

Fig 5.2.1: *Get Graduate – Application UI*

After scraping the final data will look like

```
In [39]: df.head()
```

Out[39]:

	Unnamed: 0	Unnamed: 0.1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0	0	University Of Waterloo	Systems Design Engineering	MS	NaN	Accepted	Website	(1, 7, 2019)	1.561964e+09	NaN	NaN	NaN	NaN	NaN	NaN	International
1	1	1	Northeastern University	Electrical Engineering	PhD	F19	Rejected	Website	(8, 7, 2019)	1.562569e+09	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	2	2	The University Of Auckland	Electrical And Electronic Engineering	MS	NaN	Accepted	Website	(19, 6, 2019)	1.560928e+09	NaN	NaN	NaN	NaN	NaN	NaN	International
3	3	3	Radford University	Counseling Psychology PsyD.	Other	F19	Accepted	Phone	(4, 3, 2019)	1.551686e+09	NaN	NaN	NaN	NaN	NaN	NaN	American
4	4	4	University Of Chittagong	Computer Science	MS	NaN	NaN	Other	(9, 7, 2019)	1.562656e+09	3.2	163.0	168.0	4.0	True	NaN	International

Fig 5.2.2: *Get Graduate – Application UI*

The list of attributes are made as a dataset for pre-process cleansing. For graduate students the dataset consists of University Name, Major, Degree, Season, Decision, Decision Method, Decision Date, Undergraduate GPA, Is New GRE Verbal, GRE Quant, GRE Writing, Status, Postdate Comments, Research Experience, Recommendations and Undergraduate GPA. For Under graduate students the dataset consists of Student profile and SAT scores.

Undergraduate student dataset:

Undergraduate student data is taken from the College rank score card website https://collegescorecard.ed.gov/data/. The data before cleaning looked like below.

UNITID	OPEID	OPEID6	INSTNM	CITY	STABBR	ZIP	ACCREDAGEI	INSTURL	NPCURL	SCH_DEG	HCM2	MAIN	NUMBRANCI	PREDDEG	HIGHDEG	CONTROL	ST_FIPS	REGION
100654	100200	1002	Alabama A &	Normal	AL	35762	NULL	NULL	NULL	NULL	NULL	1	1	3	4	1	1	5
100663	105200	1052	University of	Birmingham	AL	35294-0110	NULL	NULL	NULL	NULL	NULL	1	1	3	4	1	1	5
100690	2503400	25034	Amridge Uni	Montgomery	AL	36117-3553	NULL	NULL	NULL	NULL	NULL	1	1	3	4	2	1	5
100706	105500	1055	University of	Huntsville	AL	35899	NULL	NULL	NULL	NULL	NULL	1	1	3	4	1	1	5
100724	100500	1005	Alabama Sta	Montgomery	AL	36104-0271	NULL	NULL	NULL	NULL	NULL	1	1	3	4	1	1	5
100751	105100	1051	The Universi	Tuscaloosa	AL	35487-0166	NULL	NULL	NULL	NULL	NULL	1	1	3	4	1	1	5
100760	100700	1007	Central Alab	Alexander Ci	AL	35010	NULL	NULL	NULL	NULL	NULL	1	1	2	2	1	1	5
100812	100800	1008	Athens State	Athens	AL	35611	NULL	NULL	NULL	NULL	NULL	1	1	3	3	1	1	5
100830	831000	8310	Auburn Univ	Montgomery	AL	36117-3596	NULL	NULL	NULL	NULL	NULL	1	1	3	4	1	1	5
100858	100900	1009	Auburn Univ	Auburn	AL	36849	NULL	NULL	NULL	NULL	NULL	1	1	3	4	1	1	5
100937	101200	1012	Birmingham	Birmingham	AL	35254	NULL	NULL	NULL	NULL	NULL	1	1	3	3	2	1	5

Fig 5.2.3: *Get Graduate – Application UI*

5.3.Analysis:

Logic of creating any media through which people can get their university nearest recommendations based on their personal information containing their exam scores and other required information will be better and also will increase the chances of optimizing money on spending admit money on the suggested colleges only as we got to know from the logic of any media. Media through any web based application would be even better as accessibility for everyone will be increased and gathering recommendation information will also become easy for the users of that particular application.

Data Preprocessing:

In order to use the obtained data for our analysis, we need to do the preprocessing and cleansing, as there are lots of anomalies in the dataset. For this we use pandas and numpy frameworks.

Cleansing the data was done by

- Removing the irrelevant columns by using the drop column feature
- Filling the null values with the appropriate value or deleting the row containing null values.
- Removing the spaces in the data and reducing the size of the dataset.

In our graduate dataset, The GRE scores were additionally scrubbed since they contained scores of both old and new forms of the assessment. Likewise the GPA scores accessible depended on various point frameworks, so all the GPA scores were consistently scaled to 4 point scale by using normalized functions.

$$x \text{ normalized} = (x - x \text{ minimum}) / (x \text{ maximum} - x \text{ minimum})$$

Where, x is the value of the GPA

Exploratory data Analysis:

Exploratory data Analysis is a technique, which employs a variety of techniques. It consists of various techniques like below.

1. Plotting raw data
2. Plotting simple statistics such as mean , standard deviation, etc.
3. Positioning such plots, so as to maximize our natural pattern recognition.

This is the description of the data.

```
data.describe()
```

	decdate_ts	cgpa	greV	greQ	greA	gre_subject	post_timestamp
count	6.145400e+04	55589.000000	61474.000000	61474.000000	61474.000000	7175.000000	6.147400e+04
mean	1.431551e+09	3.715970	231.556333	248.826447	4.144757	796.411150	1.431763e+09
std	9.540728e+07	0.506153	174.575147	208.551820	1.111126	122.305977	8.079993e+07
min	-1.000000e+00	0.400000	130.000000	130.000000	0.000000	310.000000	1.263283e+09
25%	1.363244e+09	3.520000	155.000000	157.000000	3.500000	710.000000	1.363417e+09
50%	1.426662e+09	3.750000	161.000000	164.000000	4.000000	800.000000	1.427094e+09
75%	1.490771e+09	3.900000	167.000000	170.000000	5.000000	890.000000	1.491030e+09
max	1.360120e+10	9.990000	800.000000	800.000000	6.000000	990.000000	1.562569e+09

Fig 5.3.1: *Get Graduate – Application UI*

The Processed data will look like below.

```
data.columns = ['univName', 'major', 'program', 'season', 'decision', 'Method', 'decdate', 'decdate_ts', 'cgpa', 'greV'
                'greA', 'is_new_gre', 'gre_subject','status', 'post_data', 'post_timestamp', 'comments']
data.head()
```

	univName	major	program	season	decision	Method	decdate	decdate_ts	cgpa	greV	greQ	greA	is_new_gre	gre_subject	status	post
0	University Of Waterloo	Systems Design Engineering	MS	NaN	Accepted	Website	(1, 7, 2019)	1.561964e+09	NaN	NaN	NaN	NaN	NaN	NaN	International	
1	Northeastern University	Electrical Engineering	PhD	F19	Rejected	Website	(8, 7, 2019)	1.562569e+09	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	The University Of Auckland	Electrical And Electronic Engineering	MS	NaN	Accepted	Website	(19, 6, 2019)	1.560928e+09	NaN	NaN	NaN	NaN	NaN	NaN	International	
3	Radford University	Counseling Psychology PsyD.	Other	F19	Accepted	Phone	(4, 3, 2019)	1.551686e+09	NaN	NaN	NaN	NaN	NaN	NaN	American	
4	University Of Chittagong	Computer Science	MS	NaN	NaN	Other	(9, 7, 2019)	1.562656e+09	3.2	163.0	168.0	4.0	True	NaN	International	

Fig 5.3.2: *Get Graduate – Application UI*

Exploratory data analysis: pair plots of GRE verbal, GRE Quantitative and GPA.

```
sns.pairplot(data, palette="husl", x_vars=["greV","cgpa","greQ"], y_vars=["greV","cgpa","greQ"], height=8)
plt.show()
```

Fig 5.3.3: *Get Graduate – Application UI*

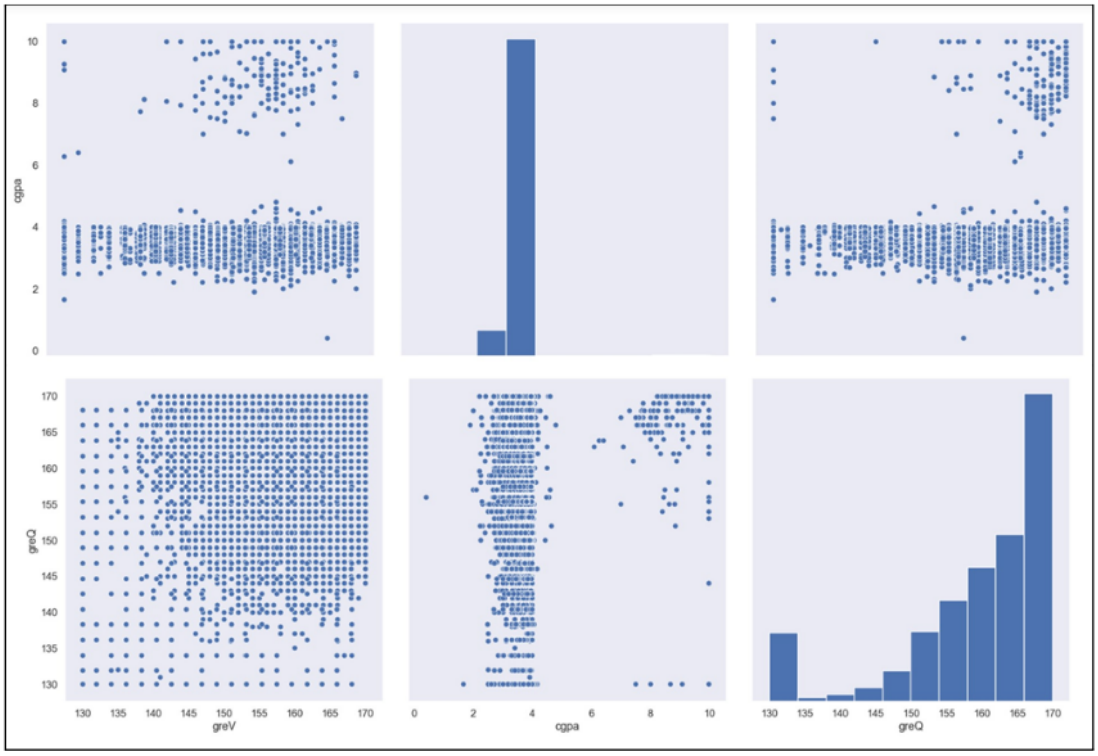

Fig 5.3.4: *Get Graduate – Application UI*

Undergraduate Data EDA:

In Undergraduate data, we have taken below a few rows of data like Institution name, city, tuition Fees, Sat Score, Admission rate, Debt and Men Ratio.(INSTNM', 'CITY', 'STABBR' ,'TUITIONFEE_OUT', 'SAT_AVG_ALL', 'ADM_RATE_ALL', 'DEBT_MDN_SUPP', 'UGDS_MEN']).

This data will be used for training the model and test data as SAT score and Maximum tuition fees.

Recommendation Methodology:

Here we have used a Knowledge based recommendation System where User inputs are taken into account and compared with the training data.

⇒For Graduate University Recommendation we have used Case based knowledge recommendation as it will take the User inputs and compare with trained data.

⇒For the Undergraduate Recommendation System, we have used a Constraint based Knowledge recommendation system where user inputs are taken into account as constraints and based on the constraints we compared with trained data.

We used two different models like K-Nearest Neighbors for Graduate data and Feature weighted algorithms for Undergraduate data.

K Nearest Neighbor:

In KNN, the trained data is compared with test data and distances are calculated using Euclidean distance. It then classifies an instance by finding its nearest neighbors and recommends the top n nearest neighbor universities.

Algorithm is stated below.

Input: undergraduate university, department, CGPA, GRE Scores of User

1. Initialize the value of k
2. For getting recommendation, iterate from 1 to number of trained data
3. Calculate distance between test data and each row in the trained data.
4. Sort the distances in ascending order
5. Get top k rows and recommend to the user

Output: strongly suggested N Outgoing University examining Universal Table of Previous Successful Students will be recommended

Flowchart of the graduate recommendation System:

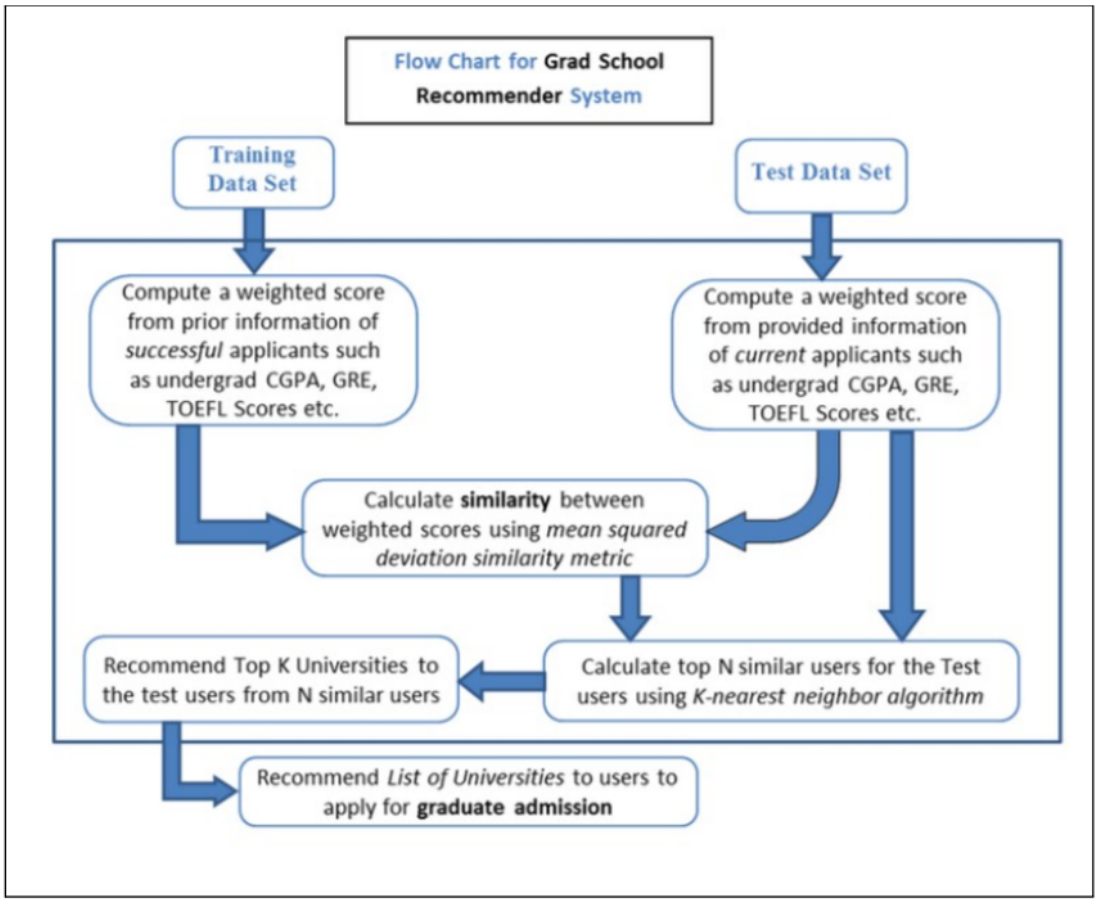

Fig 5.3.5: *Get Graduate – Application UI*

IMPLEMENTATION

Training data for the KNN algorithm:

	univName	cgpa	greV	greQ	greA
14	Ohio State University	4.00	150.0	166.0	3.0
17	Texas A&M University	3.57	157.0	151.0	5.5
46	University Of California, Irvine	3.66	155.0	167.0	4.0
64	Boston University	3.10	161.0	157.0	4.0
203	Oregon State University	3.38	154.0	170.0	4.0

Fig 5.3.6: *Get Graduate – Application UI*

```
def knn(trainingSet, testInstance, k):
    print(k)
    distances = {}
    sort = {}
    length = testInstance.shape[1]

    for x in range(len(trainingSet)):

        dist = euclideanDistance(testInstance, trainingSet.iloc[x], length)

        distances[x] = dist[0]

    sorted_d = sorted(distances.items(), key=lambda x: x[1])


    neighbors = []

    for x in range(k):
        neighbors.append(sorted_d[x][0])

    classVotes = {}

    for x in range(len(neighbors)):
        response = trainingSet.iloc[neighbors[x]][-1]
        if response in classVotes:
            classVotes[response] += 1
        else:
            classVotes[response] = 1

    sortedVotes = sorted(classVotes.items(), key=lambda x: x[1], reverse=True)

    return(sortedVotes, neighbors)
```

Fig 5.3.7: *Get Graduate – Application UI*

Implementation of Feature weighted algorithm for Undergraduate universities:

The weightage of all the features are taken and find the similarity score. Based on the similarity score, the universities with highest similarities will be recommended to students. Suppose w1, w2 are weights and f1 and f2 are features the similarity is calculated by formula Similarity score = w1* f1+w2*(1-f2)

Algorithm is stated below.

Input: SAT Score and Maximum tuition fees of User

1. For getting recommendation, iterate from 1 to number of trained data
2. Find the rows in the training data similar to the user provided SAT score and tuition fees.
3. Calculate the weightage of both the attributes and calculate the score as acceptance rate
4. Sort the distances in ascending order
5. Get top k rows and recommend to the user

Output: Top 5 Recommended Universities

5.4.Solution:

Based on the analysis's thought, we built an application called **FLY EASY**. This application is developed in two layers i.e. front-end and back-end.

As we are working in a group on this project, we decided to work equally and contributed accordingly so that everyone knows what others do and why.

To make you understand how we designed our web application in both the front and back end. We explained how the whole application works in a detailed way below.

5.4.1.Back-end:

In the backend part, we used FastAPI (Python) and MongoDB for managing and storing data. We built several CRUD (create, read, update, delete) operations to create and manage data. These CRUD operations are named differently in the REST services context. They are defined as

GET (read), POST (create), PUT (update), DELETE (delete)

Creating users and managing their information requires models/tables that store data with certain attributes. We created a few models/tables to store different data. They are:

- **Users:**

  Users is a collection of user details to be stored in a database.

  Attributes in users:

  username, password, firstName, lastName, dateOfBirth, greV, greQ, greA, CGPA, SAT, displayPicture

  BackEND Service Outlet:

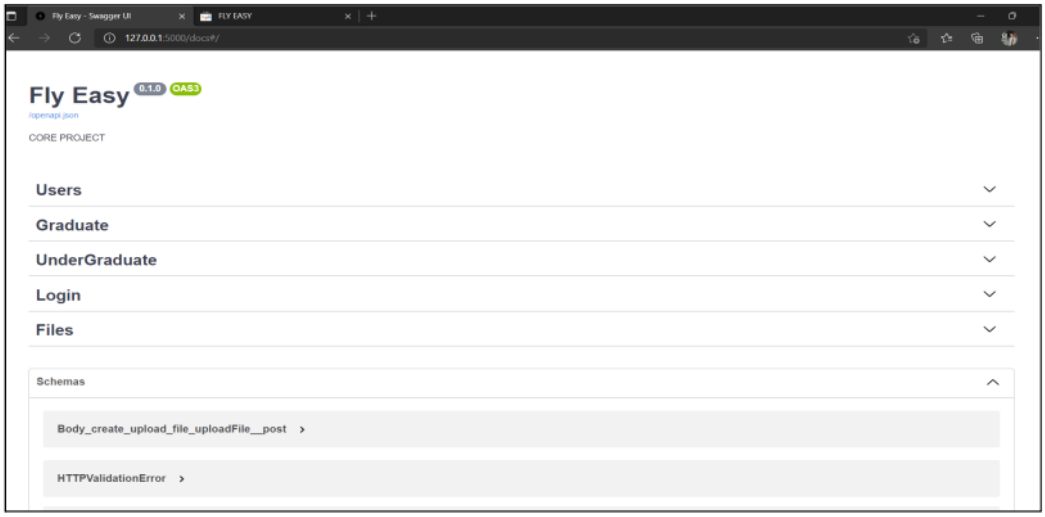

Fig 5.4.1.1: *BackEND Services Outlet*

CRUD operations:

Fig 5.4.1.2: *FastAPI user-related services*

- ## UnderGraduate:

  Recommendation based on certain inputs to suggest undergraduate universities to the user

  Attributes in Tasks:

  Mainly, SAT score and Tuition Fee in $ (estimated by user, need not to be accurate) are required ones

  UnderGraduate related API:

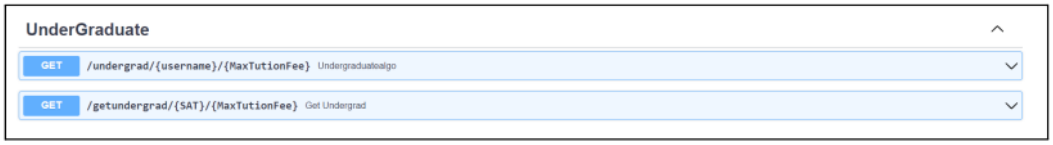

Fig 5.4.1.3: *FastAPI UnderGraduate-related services*

- ## Graduate:

  Recommendation based on certain inputs to suggest graduate universities to the user

  Attributes in Tasks:

  Mainly, scores of GRE examinations are required. So, greV(Verbal score), greQ(Quantitative score), greA(Analytical score), CGPA(grade of the student[user]) are required

  Graduate related API:

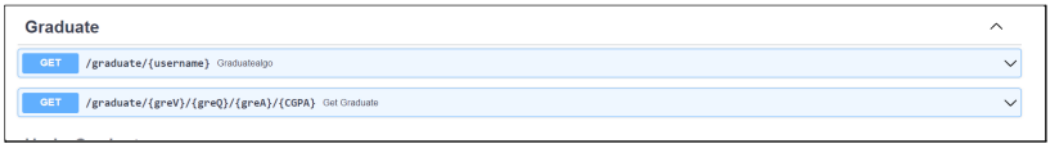

Fig 5.4.1.4: *FastAPI Graduate-related services*

Other Required Services:

We also implemented a few other required services in the backend, one to check if the login part is validating correctly or not and the other for display picture attributes so as to upload or choose the image for our respective display picture while using the application. We made the chosen picture format to be converted into binary form so as to save the image as it is in the database and to be reflected the same on the frontend part to a particular logged in user the same image he/she chose.

Fig 5.4.1.5: *FastAPI Other required services*

5.4.2.Front-end:

We made a user interface using angular in a modern way that users are familiar to operate with i.e. understandability and readability so that they can understand where to go and what to choose in the application. Some of our UI faces are presented below.

- **Login/Sign-Up Page:**

  Our login/sign-up page has a simple and modern look and the background image will be changing automatically in a small interval of time throughout the application.

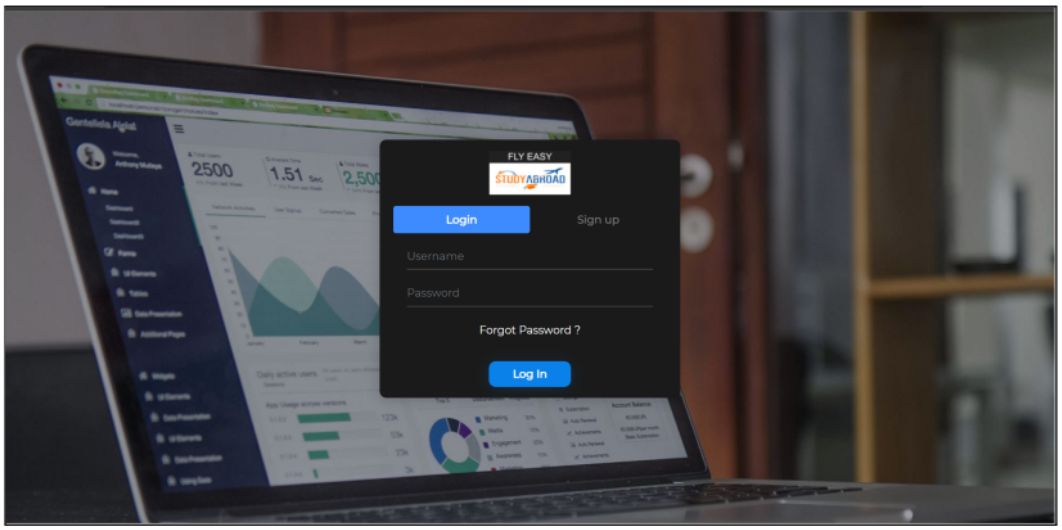

Fig5.4.2.1: *Login Page*

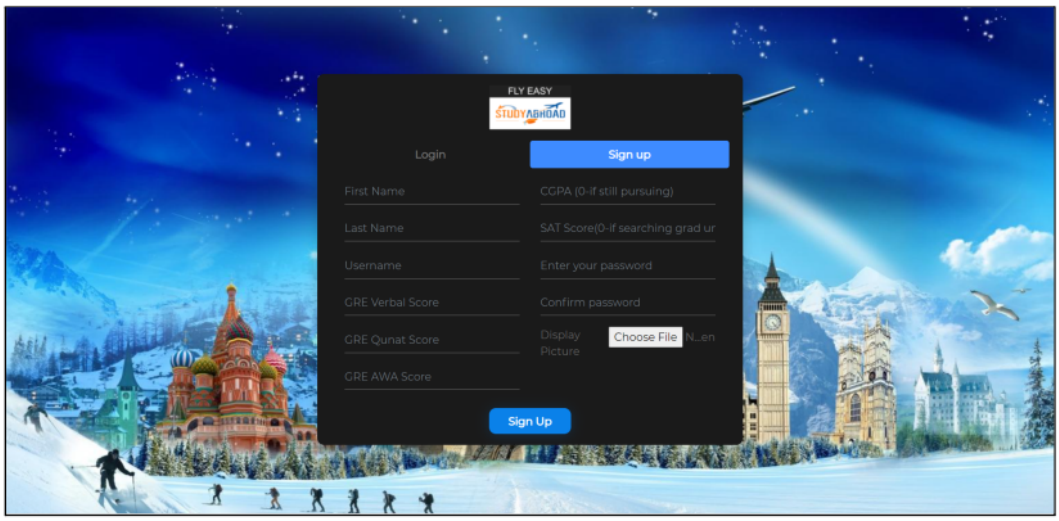

Fig 5.4.2.2: *Sign-up Page*

- **Index Page:**

  After logging in, the user will be redirected to the index page.

  In index, all activities that need to be performed by him/her and the activities that need to be done or can be done in the application are displayed.

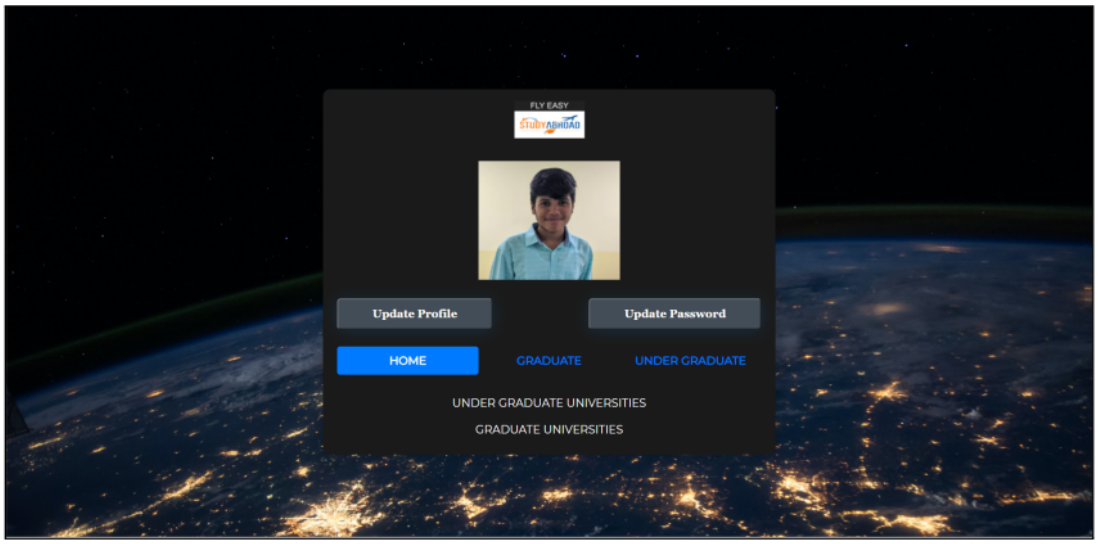

Fig 5.4.2.3: *Index Page with home bar form*

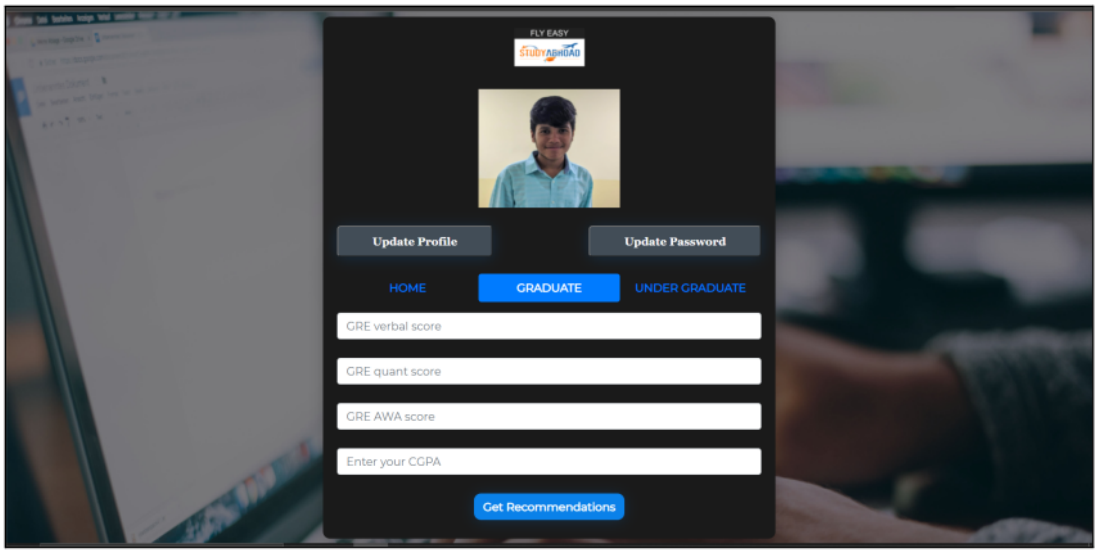

Fig 5.4.2.4: *Graduate bar form*

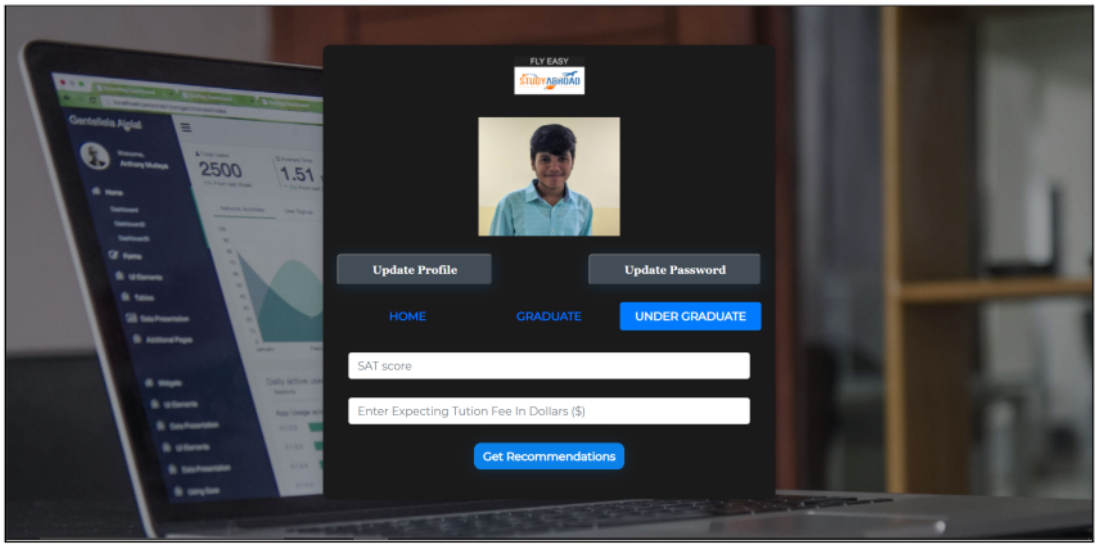

Fig 5.4.2.5: *Under Graduate bar form*

- **Update Profile Page:**
  This page allows the user to modify any of his/her own scores's information other than that they mentioned while signing up.

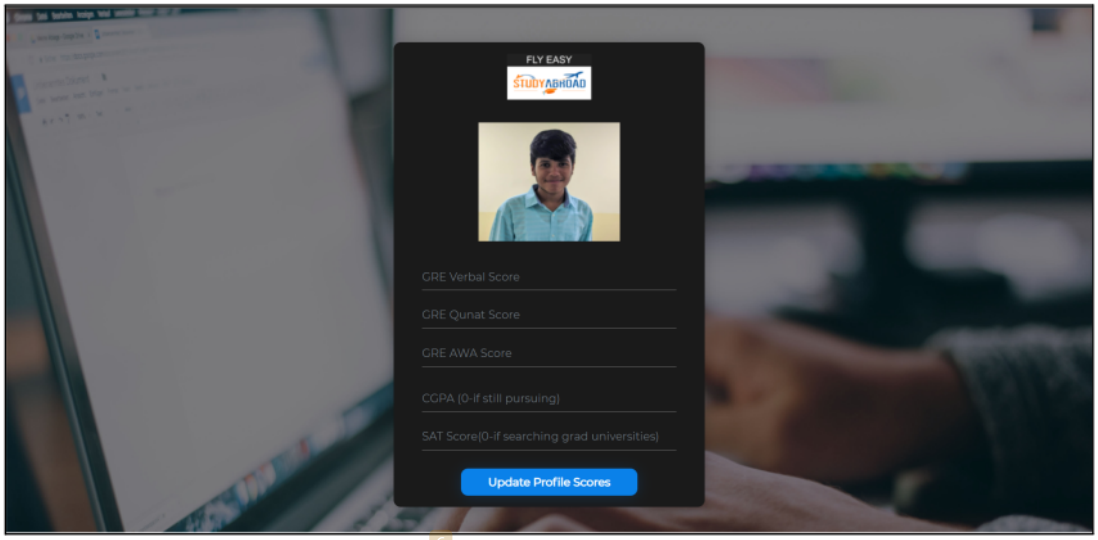

Fig 5.4.2.6: *Update Profile Scores page*

- **Update Password Page:**
  This page allows the user to modify his/her own existing password to a new password other than that they mentioned while signing up.

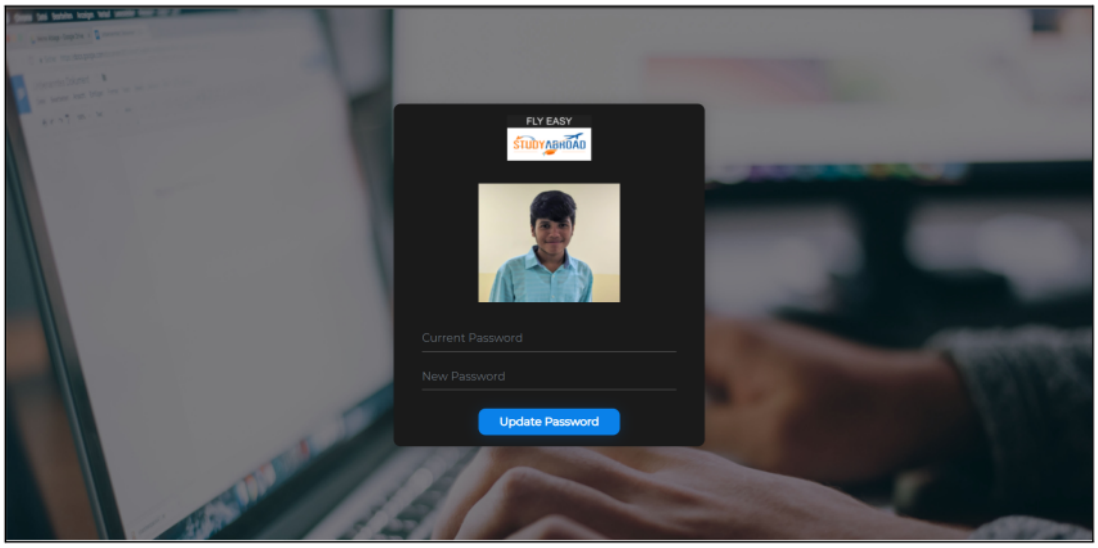

Fig 5.4.2.7: *Update Password Page*

5.4.3.Control Flow:

After knowing about the front-end and back-end of our project, we need to know how the two layers work together to respond to the user's input/request. Let's break it down into steps to understand it better.

I. Firstly, when a user performs an action like clicking a button it creates a request/service call to the backend FastAPI services.

II. Based on the request, the service that gets called, will fetch data from the database accordingly.

III. Database returns the data in the form the service requests from it.

IV. After receiving the data, the service processes the data and sends a response to the front-end. It may be either a message or data that is requested.

Below is the block diagram for the control flow we followed and implemented to design a responsive web application.

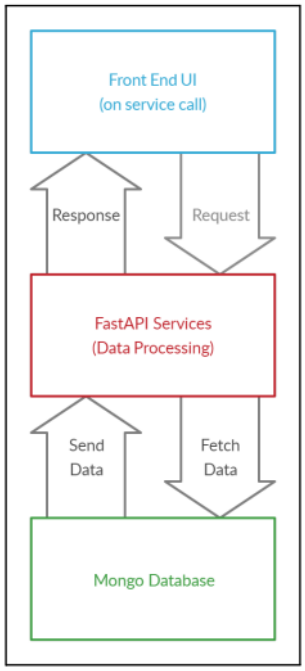

Fig 5.4.3.1: *Control flow of the application*

Flowchart also followed for graduate recommendation to the user same as shown in the figure

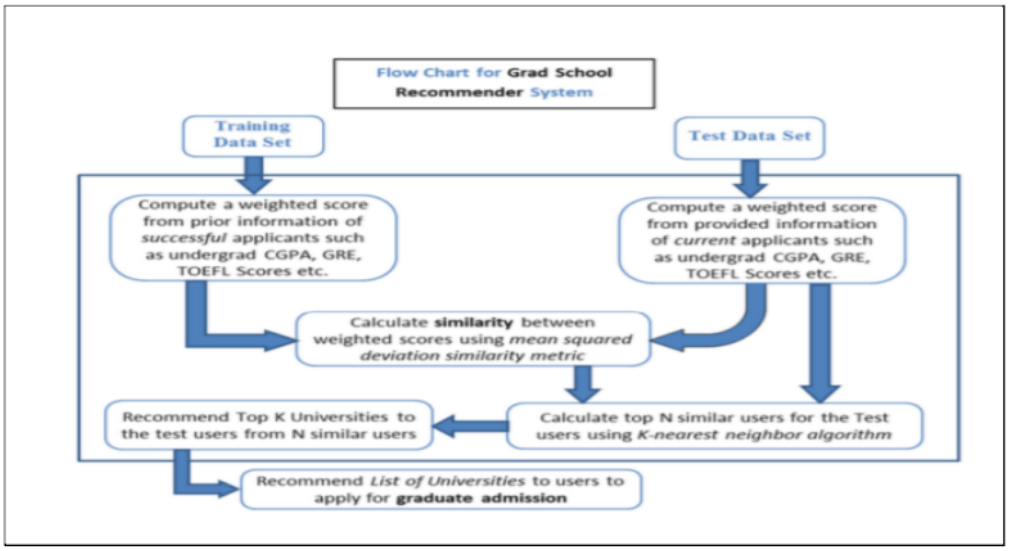

Fig 5.4.3.2: *Control flow of the Graduate Recommendation*

6.User Data Collection:

We used MongoDB for storing/collecting data. We created a core-project database and created collections like Users, file chunks and related ones for image uploadings for storing information.

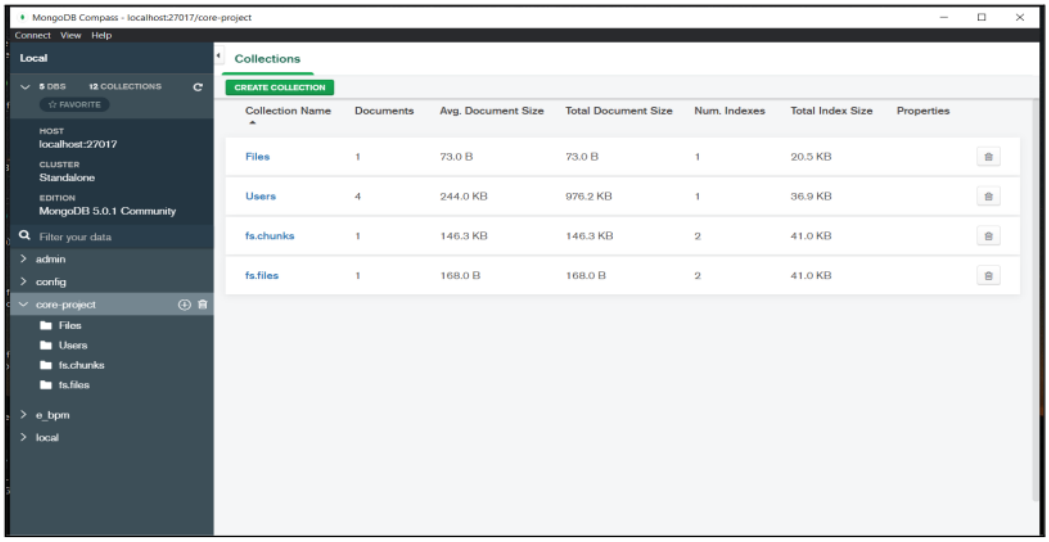

Fig 6.1: *MongoDB - Collections*

Creating and retrieving data is processed by the python services (CRUD methods) that we coded using FastAPI as a part of the backend. We can either create and retrieve data from Swagger UI (that lists all the CRUD methods) or from UI that we created which is internally connected to the same python services.

Performing undergraduate recommendation:

1) **Swagger UI**

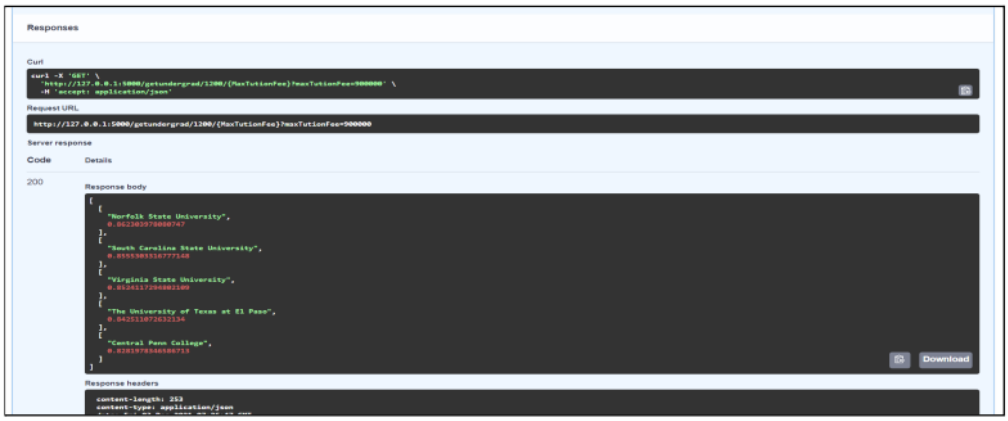

Fig 6.2: *Get UnderGraduate – Swagger UI*

7.Results and Discussion:

We are able to develop the whole functioning FLY EASY web application that overcomes all the problems that students face in selecting and applying required colleges. In a detailed way:

- Building a user interface using angular that is easy usable for users. A user can be able to check their suitable colleges for both undergraduate and graduate colleges based on their scores and also check for random scores. Also can update password and profile at their will.

- Whereas in the back-end, to retrieve data from the database many services (create, read, write, update) are created so that the data is processed accordingly. All the passwords are encrypted on a high level so that it cannot be exposed. These services are built using FastAPI (Python) that can retrieve data as fast as possible.

Undergraduate Results:

Once the user clicks on the Undergraduate button, then needs to fill in the SAT score and the maximum tuition fee and click on the undergraduate button they will get a list of 5 recommended colleges based on their profile evaluated by KNN algorithms. Below you could see we have entered a SAT score of 1300 and a maximum tuition fee of 95000 to demonstrate.

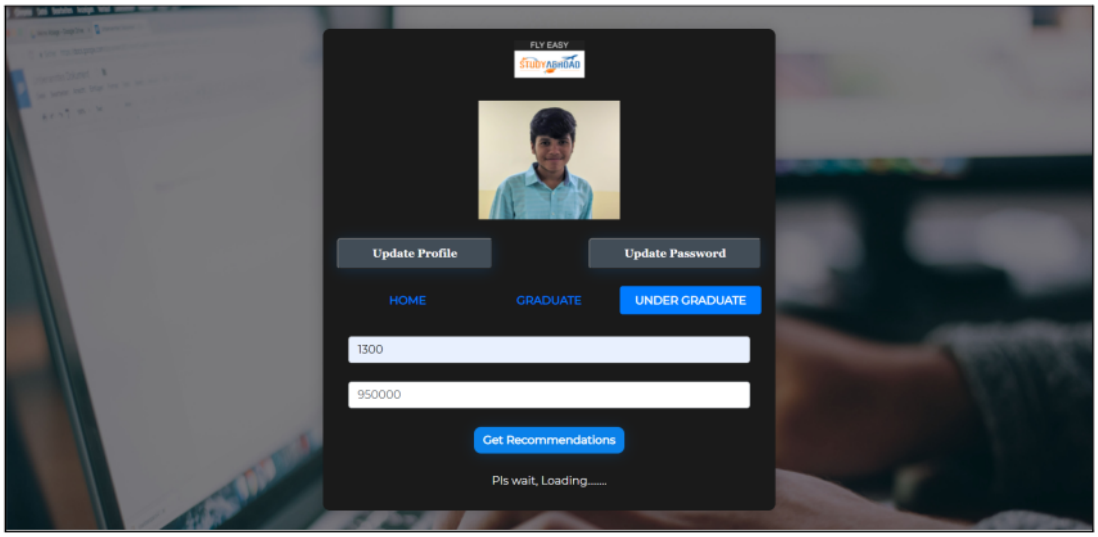

Fig 7.1: *Get UnderGraduate – Application UI*

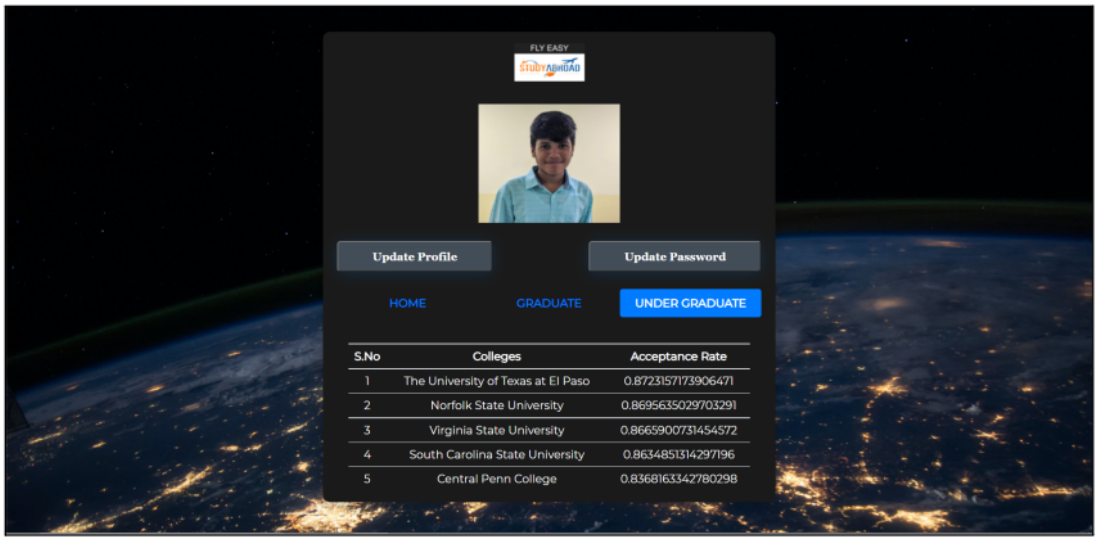

Fig 7.2: *Get UnderGraduate – Result Outlet - Application UI*

Graduate Results:

Once a user clicks on the Graduate button, then needs to fill their scores of GRE and their CGPA and the maximum tuition fee and click on the Graduate button they will get a list of 5 recommended colleges based on their profile evaluated by KNN algorithms. Below we have entered a GRE score of verbal as 150, quant as 160 and AWA of 4 and CGPA of 8.5 to demonstrate.

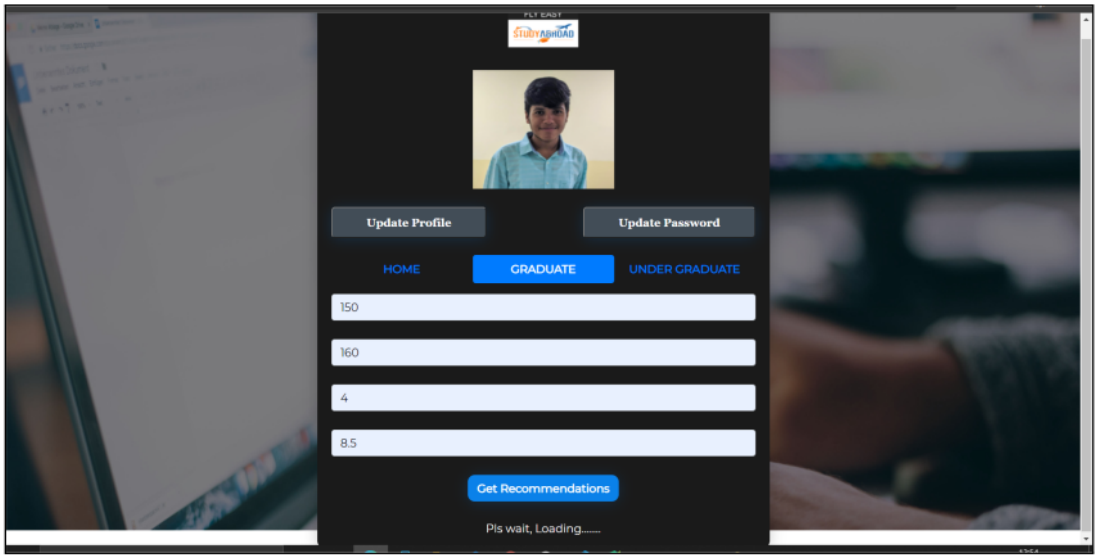

Fig 7.3: *Get Graduate – Application UI*

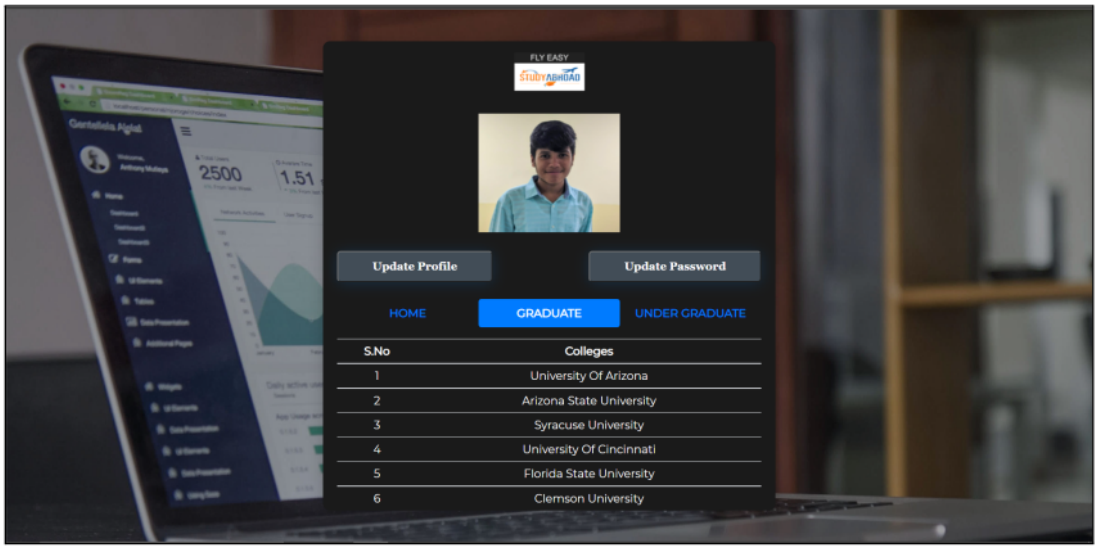

Fig 7.4: *Get Graduate – Application UI*

8.Conclusion:

This project helps students in the decision making of the universities in which they apply. The data of the previous successful applicants can be taken into account.For international admissions seekers, data from applicants' academic histories is quite crucial. In this study, we devised a method for exploiting successful applicants' academic records to create FLY EASY, a college recommender system that can assist current admissions applicants. To begin, we use weighted scores to determine how comparable the training and test data sets are. Earlier data from fruitful candidates, for example, student CGPA, GRE, TOEFL scores, and any remaining relevant records accessible in the general information base, is utilized to create the weighted scores. In order to calculate top N comparable colleges and subsequently recommend top K institutions to the users, we employed the K Nearest Neighbor algorithm for graduate universities and a feature weighted method for undergraduate colleges.Our project FLY EASY, a college recommender system, would provide a recommended list of colleges and universities to candidates based on their profile interested in pursuing higher education overseas and will aid them in applying for graduate admission at relevant institutions.

9.References:

1. https://www.semanticscholar.org/paper/Recommender-System-for-Graduate-Studies-inUSA-Suresh/22924fda3f293f80a3f62f32799c08d0b81a9b20
2. https://www.researchgate.net/publication/311758642_Graduate_school_recommender_system_Assisting_admission_seekers_to_apply_for_graduate_studies_in_appropriate_graduate_s chools
3. http://jmcauley.ucsd.edu/cse258/projects/fa15/026.pdf
4. https://devpost.com/software/graduate-school-recommendation-system

[Turnitin]CoreProject_Report_FLY-EASY

ORIGINALITY REPORT

5%	1%	3%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Mahamudul Hasan, Shibbir Ahmed, Deen Md. Abdullah, Md. Shamimur Rahman. "Graduate school recommender system: Assisting admission seekers to apply for graduate studies in appropriate graduate schools", 2016 5th International Conference on Informatics, Electronics and Vision (ICIEV), 2016 Publication	2%
2	Damar Zaky, P. H. Gunawan. "Computational Parallel of K-Nearest Neighbor on Page Blocks Classification Dataset", 2020 8th International Conference on Information and Communication Technology (ICoICT), 2020 Publication	1%
3	Submitted to Keller Graduate School of Management Student Paper	1%
4	pinnacle.allenpress.com Internet Source	<1%
5	www.wallstreetmojo.com	

	Internet Source	<1%
6	Submitted to Gulf College Oman Student Paper	<1%
7	www.slideshare.net Internet Source	<1%
8	Submitted to University of Southampton Student Paper	<1%

Exclude quotes	On	Exclude matches	< 6 words
Exclude bibliography	On		