**UNIVERSITY ADMIT ELIGIBILITY PREDICTOR**

**A PROJECT REPORT**

Submitted by

**Abishak A(953219106002)**

**Santhosh Kumar B(953219106030)**

**Siva Kumar T(953219106035)**

**Sivanraj M(953219106036)**

*in partial fulfillment for the award of the degree of*

**BACHELOR OF ENGINEERING**

in

**ELECTRONICS AND COMMUNICATION ENGINEERING**



**UNIVERSITY VOC COLLEGE OF ENGINEERING,THOOTHUKUDI**

**ANNA UNIVERSITY:: CHENNAI 600025**

**NOV 2022**

**BONAFIDE CERTIFICATE**

Certified that this project report “University Admit Eligibilty Predictor by Applied Data Science” is the bonafide work of “ABISHAK.A (953219106002),

SANTHOSH KUMAR.B (953219106030), SIVA KUMAR.T (953219106035), SIVANRAJ.M (953219106036) who carried out the project work under my supervision.

Dr.A.MOOKAMBIGA(Assistant Professor),

Electronics and Communication Engineering,

University VOC College of Engineering,

Thoothukudi-8.

**MENTOR**

Dr.K.ESSAKI MUTHU,

Electronics and Communication Engineering,

University VOC College of Engineering,

Thoothukudi-8.

**HEAD OF THE DEPARTMENT**

**CONTENTS**

**1. INTRODUCTION**

1.1 Project Overview

1.2 Purpose

**2. LITERATURE SURVEY**

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

**3. IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

**4. REQUIREMENT ANALYSIS**

4.1 Functional requirement

4.2 Non-Functional requirements

**5. PROJECT DESIGN**

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

**6. PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

**7. CODING & SOLUTIONING (Explain the features added in the project along with code)**

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

**8. TESTING**

8.1 Test Cases

8.2 User Acceptance Testing

**9. RESULTS**

9.1 Performance Metrics

**10. ADVANTAGES & DISADVANTAGES**

**11. CONCLUSION**

**12. FUTURE SCOPE**

**13. APPENDIX**

13.1 Source Code

13.2 GitHub & Project Demo Link

**1.INTRODUCTION**

**1.1 Project Overview**

This project is created using Machine Learning and Regression methods- a statistical technique to predict the outcome of event which is to verify the users’ admission eligibility level, considering the universities they have chosen. This is achieved based on the algorithms implemented, when is user feed the application with the required information (GRE Score, SOP, Research, CGPA, etc.) the results that is, whether if the student is selected or not is displayed. The student is also provided with the visualized form of their degree of eligibility compliance with university requirements represented with graphs and plots. Rather having to calculate the eligibility by the students themselves, which is also an error prone process this application helps to provide them the ease of providing them the results based on algorithmic models.

**1.2 Purpose**

For the fresh graduates of high school who for most part is unaware of the time management techniques that causes them to make mistakes when at haste, choosing a university and doing research on its offerings to its students itself is a strenuous task to do. Though there are many application and also the official websites of the universities that provide insight about everything the universities have to offer, are is not a site that offers them perception of their own level of eligibility which will adhere to the university’s criteria. In the light of it, we have created this application implementing Regression, Data mining and Machine learning to solve this issue.

**2.LITERATURE SURVEY**

**2.1 Existing Problem**

In consideration to prior research performed in this field, the Bayesian Networks Algorithm have 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 thus predicted whether the aspiring student should be admitted to university on the basis of various scores of students. Since the comparisons are made only with students who got admission into the universities but not with students who got their admission rejected, this method will not be much adaptable nor does it provide help students to realize the field the lack in.

**2.2 References**

1) Prediction probability of getting an admission into a university using Machine Learning Publisher: IEEE

[IEEE prediction of admission](https://ieeexplore.ieee.org/document/9418279)

Author-A.Sivasangari, VShivani

Date of Publication: 06 May 2021

2) Using Data Mining Techniques to Predict Student Performance to Support Decision Making in University Admission Systems

Publisher: IEEE

[IEEE Decision Making in University Admission System](https://ieeexplore.ieee.org/document/9042216)

Author-Hanan Abdullah Mengash

Date of Publication: 19 March 2020

3) College Admission Prediction using Ensemble Machine Learning Model

Publisher: IRJET

[IRJET College Admission Prediction](https://www.irjet.net/archives/V8/i12/IRJET-V8I1266.pdf)

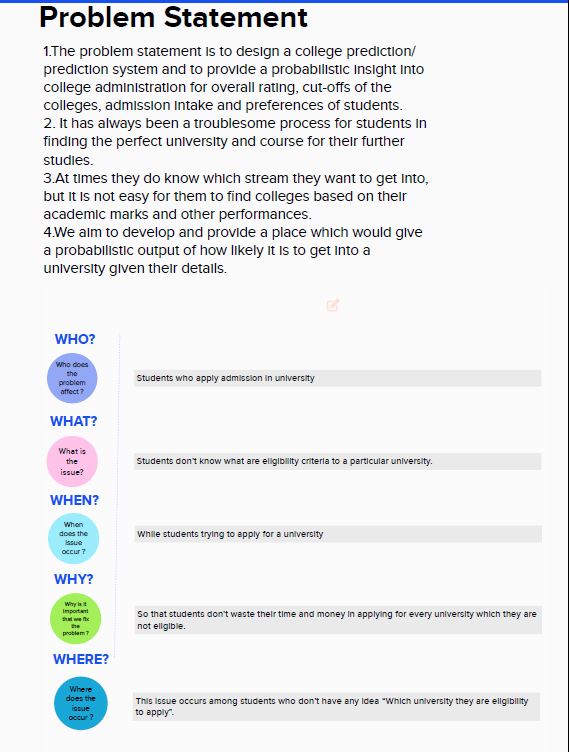
Author- Vandit Manish Jain, Rihaan Satia

Date of publication:12 December 2021

**2.3 Problem Statement Definition**

It is no secret that the pursuit of higher education does not come cheap. Unfortunately, these expenses start well before college or high school bound students set foot on the hallowed halls of their preferred institution. This gives rise to the question about the application cost. During the 2019-2020 admission cycle, 5.6 million college applications were submitted from students across the nation (NACAC, 2021) yet, the college enrollment numbers for 2020-21 academic year decreased by 4.2% from previous year.

The average college application fee in the U.S. is $50, while the elites charge higher rates. Among 62 four-year institutions with the highest application fees, the average is $77. Most colleges also require the submission of SAT, ACT, OR AP scores. But, the cost for registration for SAT or ACT could end up higher than the cost of application, yet students are still willing to take the tests even multiple times to increase the test scores.

The aforementioned stats clearly show there are students who thirst to learn and to enroll in their dream university. All these hard work and money spent would waste away if they were to ne turned down. The aim of this project is to help students with a system that could guide students and recommend best universities list and predict their admission chance in those universities according to their profile and scores.

**3.IDEATION AND PROPOSED SOLUTION**

**3.1 Empathy Map C**

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviors and attitudes. It is a useful tool to helps 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.

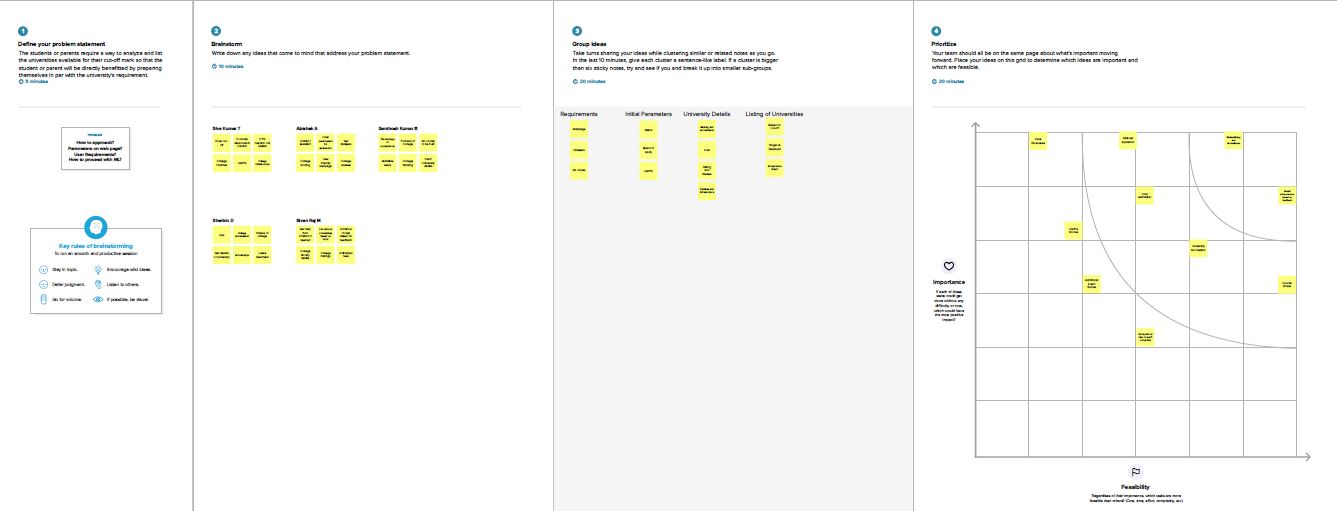


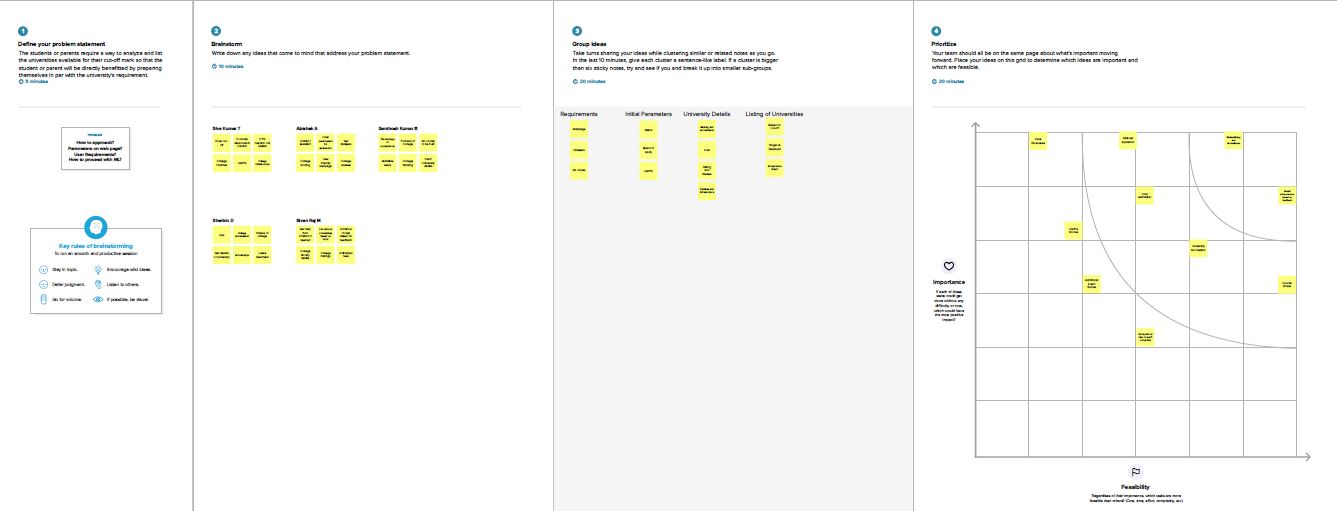
**3.2 Ideation and Brainstorming**

1.Team Gathering, Set the goal, learn to use the facilitated tools

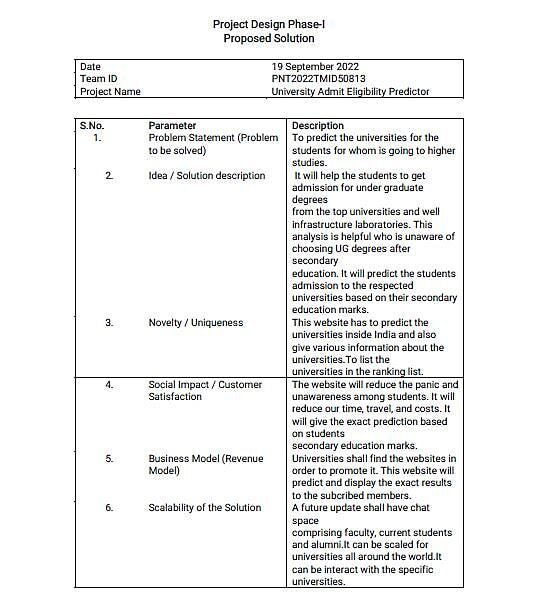
2.Brain storming and clustering ideas

3.Idea prioritization

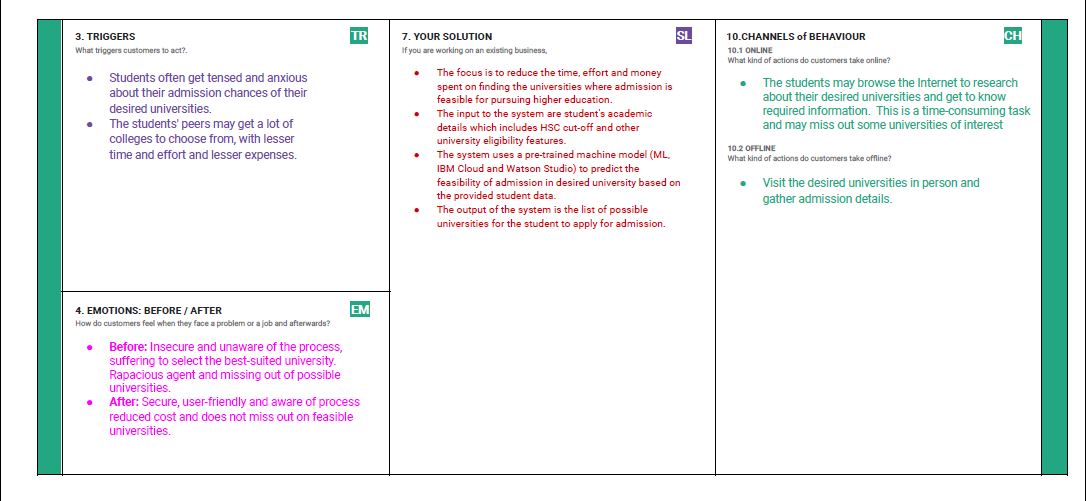
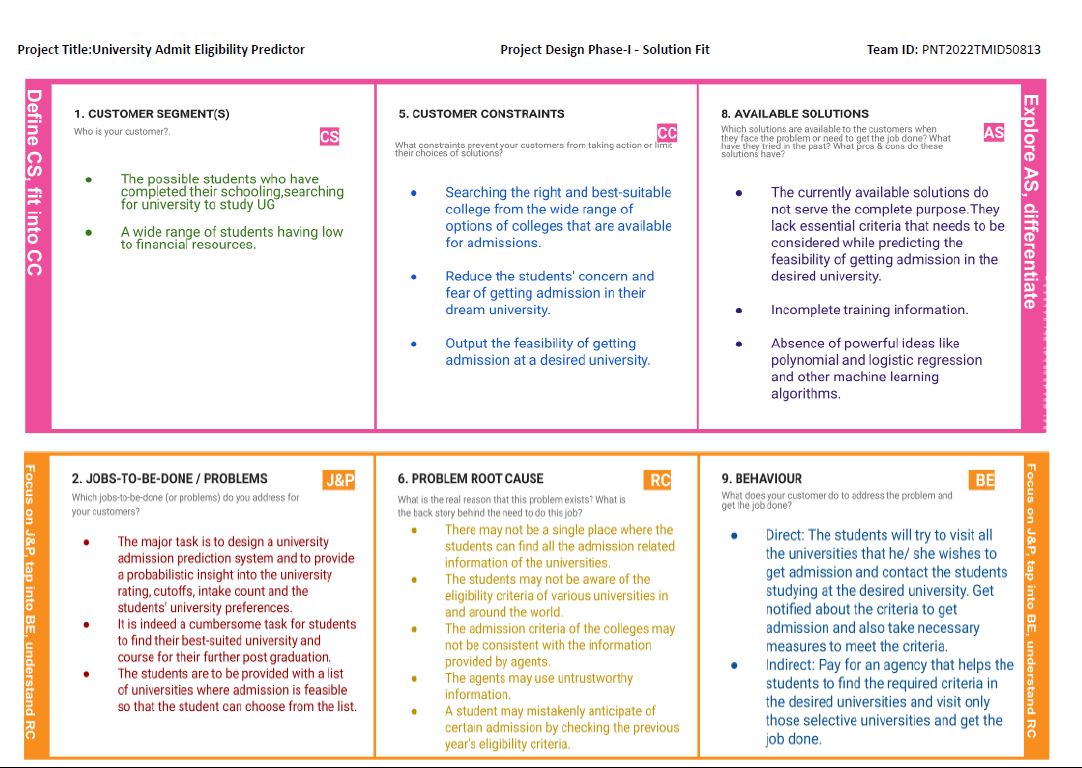




**3.3 Proposed Solution**



**3.4 Problem Solution Fit**

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer’s problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why.

**Purpose**

* Solve complex problems in a way that fits the state of your customers.
* Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
* Sharpen your communication and marketing strategy with the right triggers and messaging.
* Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
* Understand the existing situation in order to improve it for your target group.

**4.REQUIREMENT ANALYSIS**

**4.1 Functional requirement**

Following are the functional requirements of the proposed solution

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | Home Page | Description about the application.  Guidelines to use the application. |
| FR-2 | User Registration | Registration through Gmail.  Enter the required personal details. |
| FR-3 | Log in/Log out | Users can login using their mail id and password. They can logout as and when required. |
| FR-4 | User Confirmation | Confirmation via Email. |
| FR-5 | User Entry | Choose location to find universities in chosen location.  Enter academic scores. |
| FR-6 | Result | Lists the universities available by prediction based on the details entered by the user. |
| FR-7 | Resources Page | Provides information to the universities official page. Details of the universities eligibility criteria and admission process |

**4.2 Non-functional Requirements**

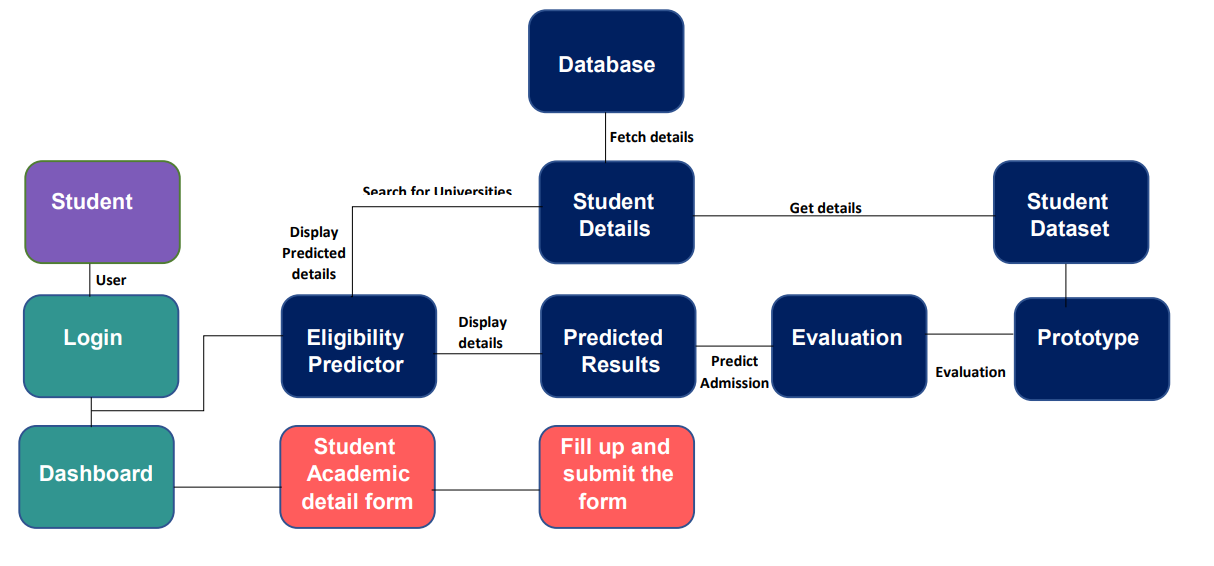
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | Usability | The application is user friendly and the admission process is made easy to understand that requires only minimal effort of the user. |
| NFR-2 | Security | The details of the are maintained confidential and are authorised. |
| NFR-3 | Reliability | The information provided are predicted with high accuracy with preferred location. |
| NFR-4 | Performance | Reduction in overall time taken to analyse data and prediction criteria with back up hence application doesn’t crash. |
| NFR-5 | Availability | Available for any user who requires details about Indian universities. |
| NFR-6 | Scalability | Supports many users at a time while maintaining optimal performance without a server crash. |

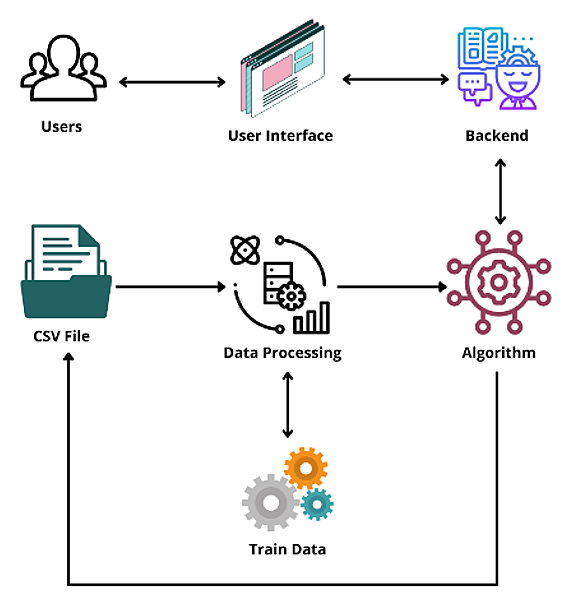
**5. PROJECT DESIGN**

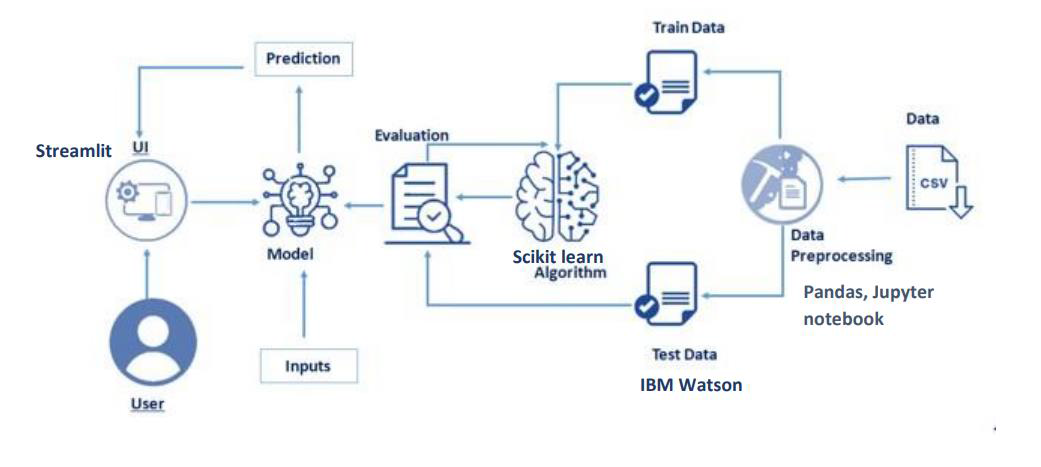
**5.1 Data Flow Diagrams**

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 enter and leaves the system, what changes the information, and where data is stored.



**5.2 Solution & Technical Architecture**

**Solution Architecture:** 

**Technical Architecture:**

**TABLE-1 : COMPONENTS & TECHNOLOGIES**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Component** | **Technology** |
| 1. | User | HTML, CSS, React Js |
| 2. | Application Logic-1 | Python |
| 3. | Application Logic-2 | IBM Watson |
| 4. | Machine Learning Algorithm | Multiple Linear  Random Forest |

**TABLE-2: APPLICATION CHARACTERISTICS**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Characteristics** | **Technology** |
| **1.** | **Open-Source Frameworks** | **Flask** |
| **2.** | **Performance** | **It handles about 100 requests per second** |

**5.3 User Stories**

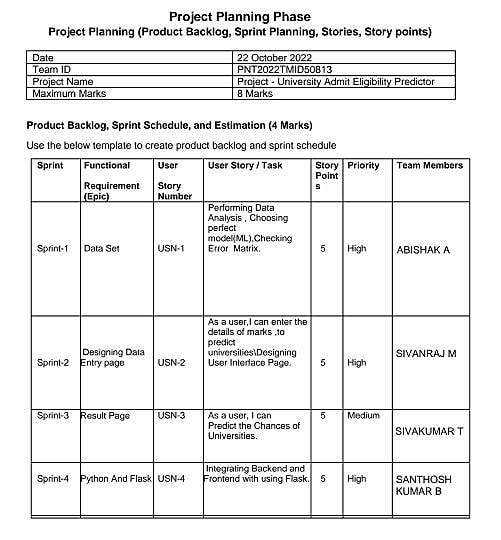
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User Story**  **Number** | **User Story / Task** | **Story Points** | **Priority** | **Team**  **Members** |
| Sprint-1 | itinerary | USN-1 | To understand using the detailed description provided | 1 | Low | Abishak A |
| Sprint-1 | Data analysis | USN-2 | To perform Performance of data visualization using matplotlib | 2 | Medium | Abishak A |
| Sprint-1 | Login/Logout | USN-3 | As a user, I can login for the application | 3 | Medium | Abishak A |
| Sprint-2 | Web development | USN-4 | To Develop a web page using stream-lit with pickle file. | 4 | High | Sivanraj M |
| Sprint-2 | Model integration | USN-5 | To perform Integration modes using regression methods | 5 | High | Sivanraj M |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-3 | Web App Hosting | USN-6 | Connect the Git-hub repo & branch to the stream-lit cloud platform and set up CI-CD to automatically deploy new changes that’s pushed to the repo. | 8 | HIGH | Siva Kumar T |
| Sprint-3 | Model deployment | USN-7 | Register in IBM cloud. Use IBM Watson ML service and IBM Watson Studio to deploy the Multiple Linear Regression Model. Test the deployment model with few examples | 5 | High | Siva Kumar T |
| Sprint-4 | Resource Page | USN-8 | Testing the application | 8 | Medium | Santhosh Kumar B |
| Sprint-4 | Results | USN-9 | As a user, I can view the results predicted by the application | 5 | Medium | Santhosh Kumar B |

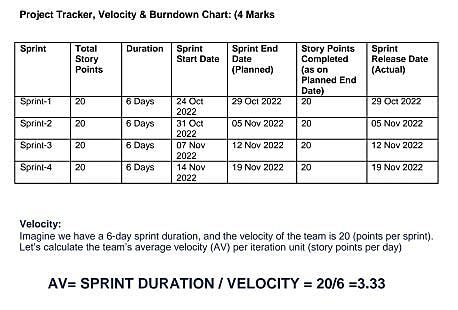
**6. PROJECT PLANNING & SCHEDULING**

**6.1 Sprint Planning & Estimation**

Product Backlog, Sprint Schedule, and Estimation

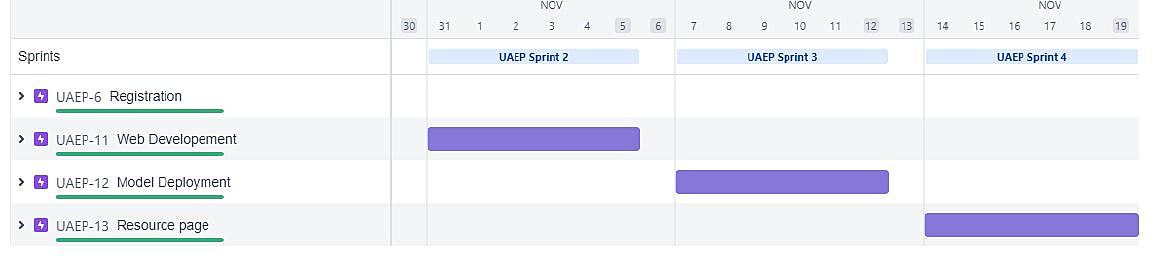


**6.2 Sprint Delivery Schedule**

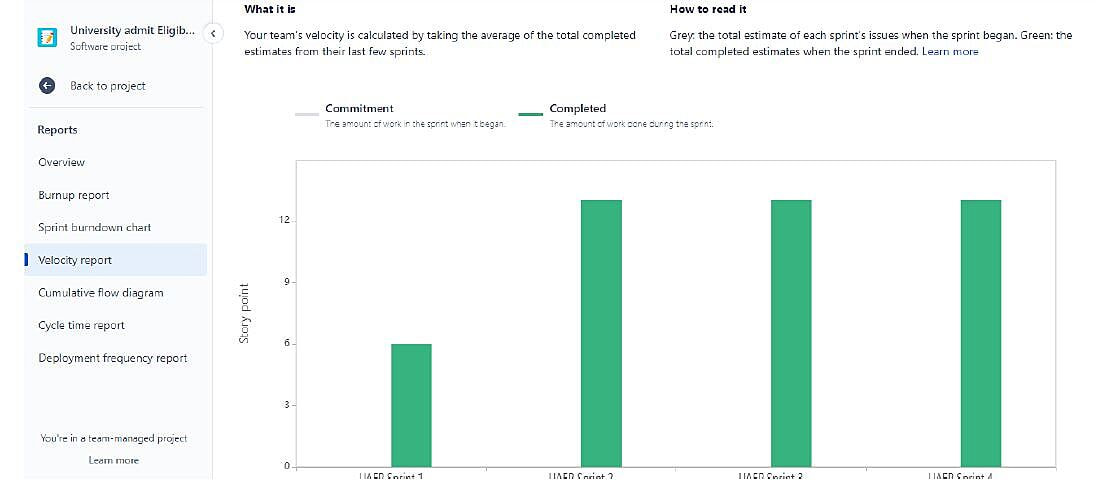


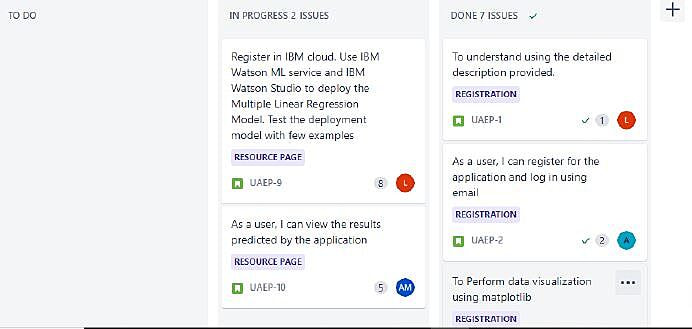
**6.3 Reports from JIRA**

**Roadmap:**

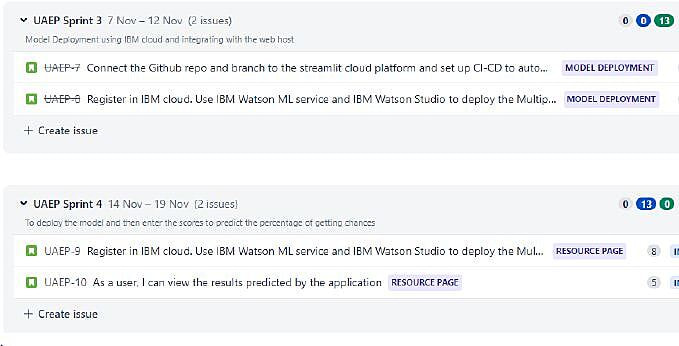
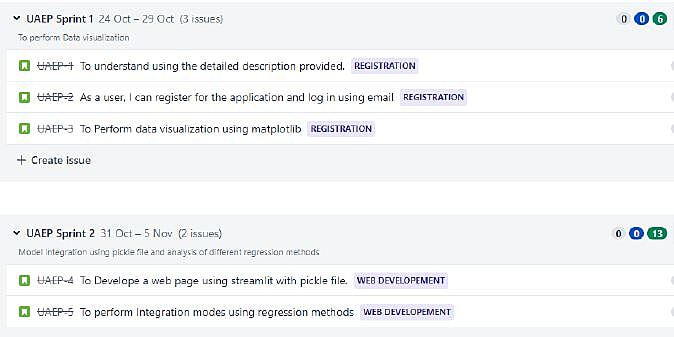


**Velocity Report:**



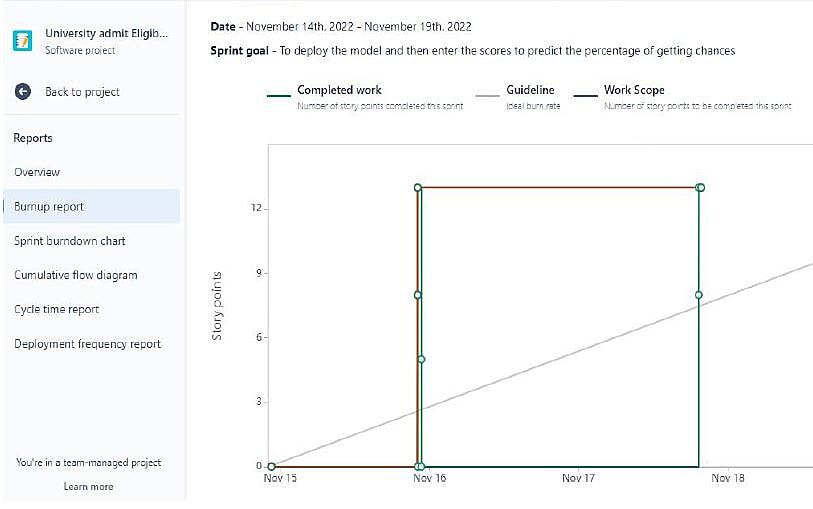
**Scrum Board:**

**Backlogs:**



**Burn down Chart:**

A burn down 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.



**7.CODING & SOLUTIONING**

**7.1 Feature 1 - UI**

The following is the UI code and working

index.html

<!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;  
 background-image: url("https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Project%20Development%20Phase/Sprint%202/static/img/index.png?raw=true");  
 background-size:cover;  
 padding: 20px 0 30px 0;  
}  
  
\* {  
 box-sizing: border-box;  
}  
  
/\* style the container \*/  
.container {  
 position: relative;  
 border-radius: 5px;  
 margin-left: 8%;  
  
}  
  
/\* style inputs and link buttons \*/  
input,  
.btn {  
 width: 200%;  
 padding: 12px;  
 border: none;  
 border-radius: 4px;  
 background-color: #d3d3d3;  
 margin: 5px 0;  
 opacity: 0.85;  
 display: inline-block;  
 font-size: 17px;  
 line-height: 20px;  
 text-decoration: none;  
}  
  
  
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:yellow;  
 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;  
}  
  
  
  
/\* hide some text on medium and large screens \*/  
.hide-md-lg {  
 display: none;  
}  
  
/\* bottom container \*/  
.bottom-container {  
 text-align: center;  
 opacity: 0.90;  
 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>ASSSS Admits - 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">  
 <div class="hide-md-lg">  
 </div>  
  
 <input type="number" name="GRE Score" placeholder="GRE Score(out of 340)" required="required" min="0" max="340"/>  
 <input type="number" name="TOEFL Score" placeholder="TOEFL Score(out of 120)" required="required" min="0" max="120"/>  
 <input type="number" name="University Rating" placeholder="University Rating(out of 5)" required="required" min="1" max="5"/>  
 <input type="number" name="SOP" placeholder="SOP(out of 5)" required="required" onkeypress="return check(event,value)" step="0.1" min="1" max="5"/>  
 <input type="number" name="LOR" placeholder="LOR(out of 5)" required="required" onkeypress="return check(event,value)" step="0.1" min="1" max="5"/>  
 <input type="number" name="CGPA" placeholder="CGPA(out of 10)" required="required" onkeypress="return check(event,value)" step="0.01" min="1" max="10"/>  
 <input type="number" name="Research" placeholder="Research(1=Yes or 0=No)" required="required" min="0" max="1"/>  
  
 <input type="submit" value="Predict"></input>  
  
 </div>  
  
 </div>  
 </form>  
  
</div>  
  
  
</body>  
</html>

chance.html

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta http-equiv="X-UA-Compatible" content="IE=edge">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Document</title>  
 <style>  
 \*{  
 margin: 0;  
 padding: 0;  
 }  
 body{  
  
 background-image: url("https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Project%20Development%20Phase/Sprint%202/static/img/chance.png?raw=true");  
 background-size: cover;  
 background-position: 25%;  
  
 background-repeat: no-repeat;  
  
 height: 100vh;  
 }  
 section{  
 /\* border:10px solid gray; \*/  
 height: 90vh;  
  
 position: relative;  
 }  
 .container{  
  
 border:10px solid gray;  
 max-width: 700px;  
 width: 100%;  
 opacity: 0.95;  
  
 height:25vh;  
 display:flex;  
 align-items: center;  
 justify-content: center;  
  
 position: absolute;  
 right: 0;  
 left: 0;  
 top: 0;  
 bottom:60%;  
 margin: auto;  
  
 }  
  
 </style>  
 </head>  
<body>  
  
 <section>  
  
 <div class="container" >  
 <h1> Prediction:{{content}}  
 <br>  
 You have a Chance </h1>  
 </div>  
 </section>  
  
  
  
  
</body>  
</html>

nochance.html

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <meta http-equiv="X-UA-Compatible" content="IE=edge">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Document</title>  
 <style>  
 \*{  
 margin: 0;  
 padding: 0;  
 }  
 body{  
 background-image: url("https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Project%20Development%20Phase/Sprint%202/static/img/no%20chance.png?raw=true");  
 background-size:cover;  
 background-position: top;  
 background-repeat: no-repeat;  
 height: 100vh;  
 }  
 section{  
 /\* border:10px solid gray; \*/  
 height: 90vh;  
  
 position: relative;  
 }  
 .container{  
 border:10px solid gray;  
 max-width: 700px;  
 width: 100%;  
 opacity: 0.90;  
  
  
 height:25vh;  
 display:flex;  
 align-items: center;  
 justify-content: center;  
  
 position: absolute;  
 right: 0;  
 left: 0;  
 top: 0;  
 bottom:60% ;  
 margin: auto;  
  
 }  
 .container{  
 font-family: Arial, Helvetica, sans-serif;  
 -webkit-text-fill-color:whitesmoke;  
 -webkit-text-stroke: 2px rgb(28, 120, 136);  
  
 }  
  
 </style>  
</head>  
<body>  
  
 <section>  
 <div class="container">  
 <h1>Prediction:{{content}}  
 <br>  
 You Do not have a Chance  
  
 </h1>  
 </div>  
 </section>  
  
  
</body>  
</html>

**7.2 Feature 2 - FLASK APP**

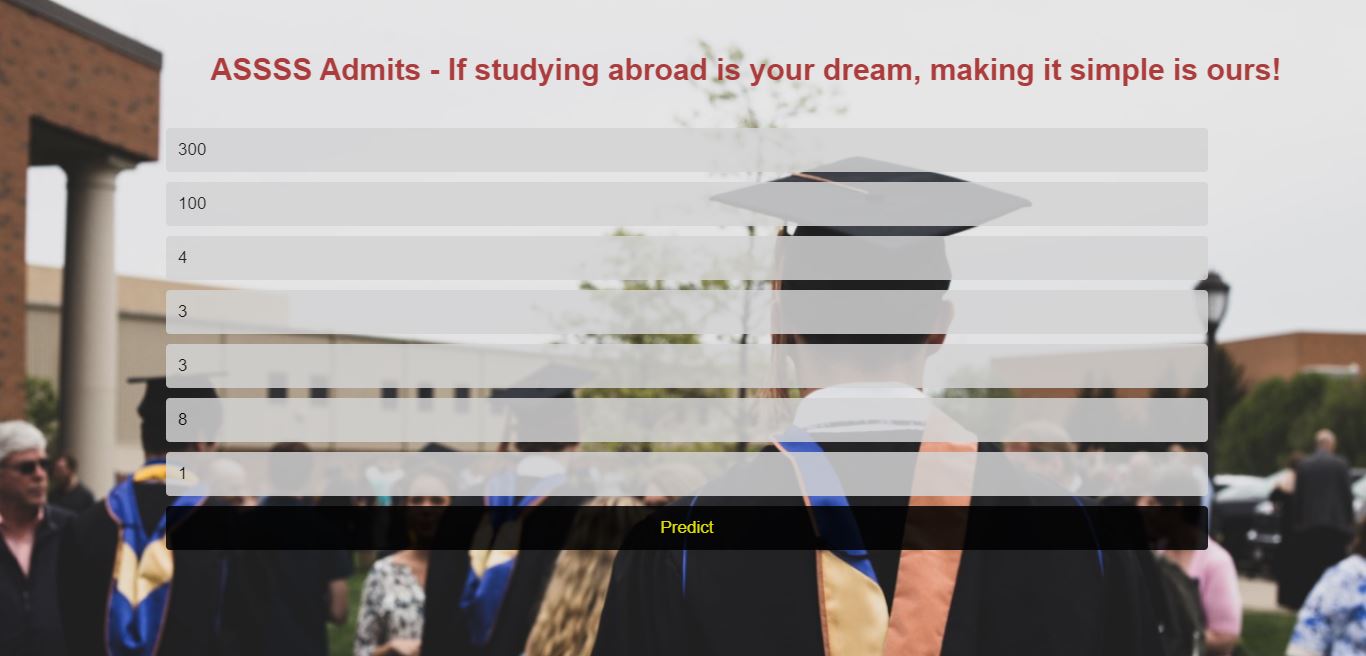
The following is the flask app code and working

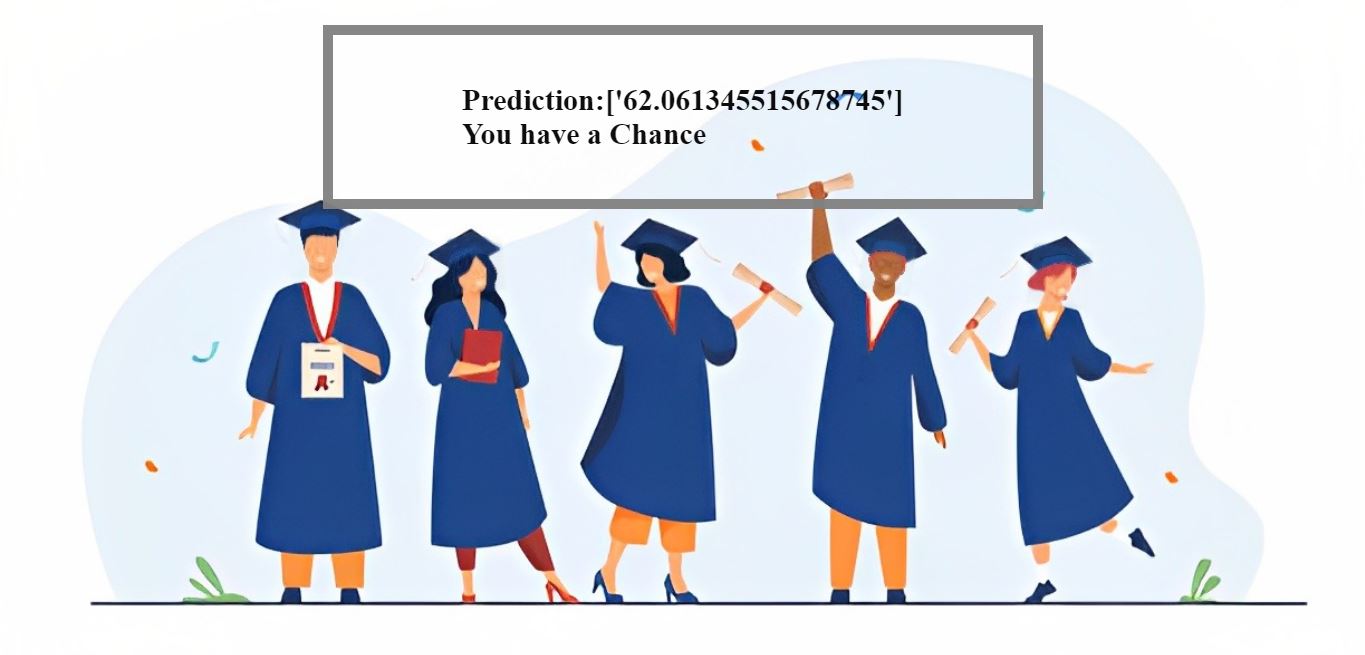
import pandas as pd  
from flask import Flask, request, jsonify, render\_template, redirect, url\_for  
import requests  
  
# NOTE: you must manually set API\_KEY below using information retrieved from your IBM Cloud account.  
API\_KEY = "wYjS7xFVjurVWF0LtAT5\_FJwgyKYVud4scNeJxF6RyQ1"  
token\_response = 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\_\_, template\_folder='templates')  
  
  
@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 = [[GRE\_Score, TOEFL\_Score, University\_Rating, SOP, LOR, CGPA, Research]]  
  
 payload\_scoring = {'input\_data': [  
 {'fields': [["GRE Score", "TOEFL Score", "University Rating", "SOP", "LOR ", "CGPA", "Research"]],  
 'values': final\_features}]}  
 print("hello")  
 response\_scoring = requests.post(  
 'https://eu-gb.ml.cloud.ibm.com/ml/v4/deployments/e6e20418-0467-4649-8bd4-cda0dd5e580a/predictions?version=2022-11-18',  
 json=payload\_scoring,  
 headers={'Authorization': 'Bearer ' + mltoken})  
 print("scoring response")  
 pred = response\_scoring.json()  
 print(pred)  
 output = pred['predictions'][0]['values'][0][0]  
 percent=output\*100  
  
 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)

**8. TESTING**

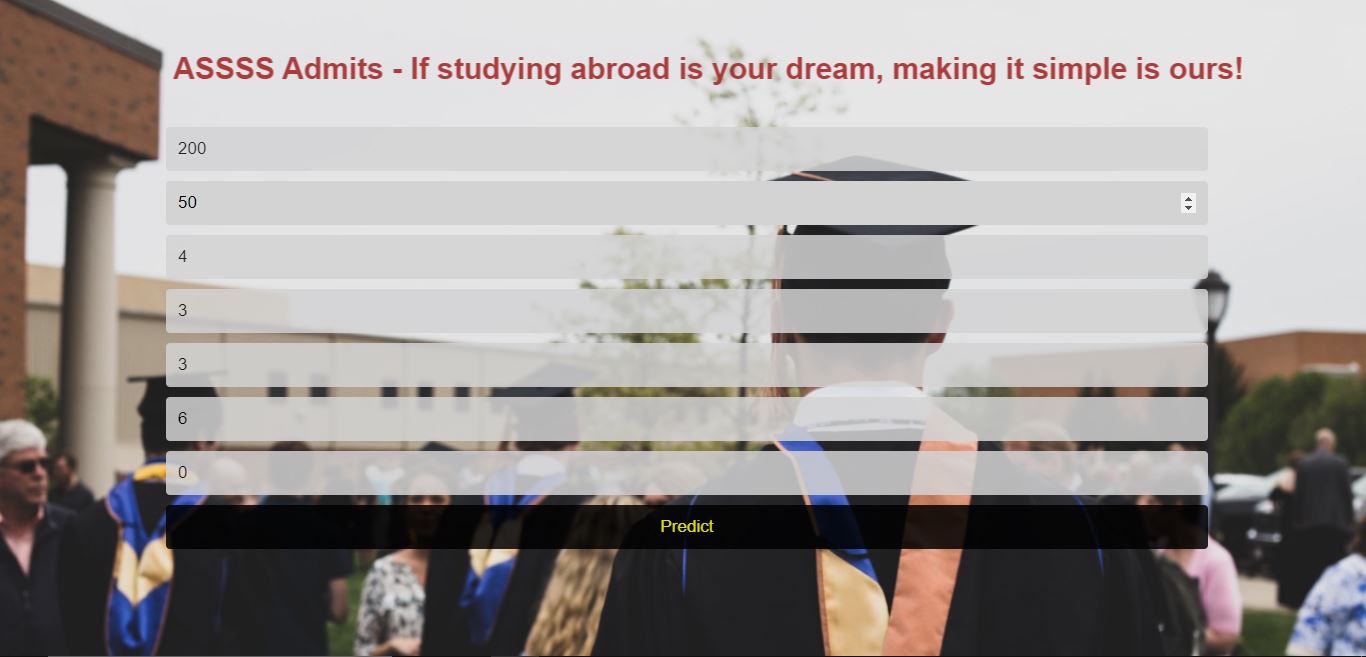
**8.1 Test Cases**

Case 1- For eligible students





Case 2- For eligible students





**8.2 Test Cases**

**User Acceptance Testing**

User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to

verify/accept the software system before moving the software application to the production

environment. UAT is done in the final phase of testing after functional, integration and system

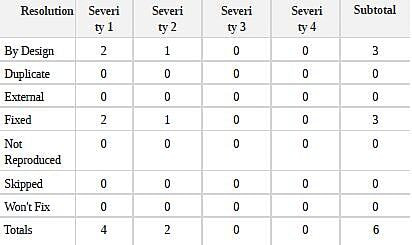
testing is done. The User Acceptance of this product is not surveyed enough to give a solid

conclusion. The theretical and hypothetical acceptance is calculated to be high enough to

conclude that this product is usable and valuable.

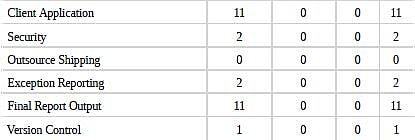
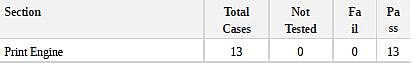
**Defect Analysis**

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



**Test Case Analysis**

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



**9.1 PERFORMANCE METRICS**

Performance Testing - Machine Learning:

|  |  |  |  |
| --- | --- | --- | --- |
| S.No. | Parameter | Values | Output |
| 1. | Metrics | Regression Model:  MAE - , MSE - , RMSE - , R2 score - | Mean Absolute Error : 0.427  Mean Square Error : 0.0037  Root Mean Square Error : 0.060  R2 Score :0.808 |
| 2. | Comparing R2 scores for different regression models | Multiple Linear Regression, Random Forest Regression | Multiple Linear Regression's Score = 0.808  Random Forest Regression's Score = 0.778 |

**10.I ADVANTAGES:**

* Avoids data redundancy and inconsistency.
* It is fast, efficient and reliable.
* It helps student for making decision for choosing a right college.
* Here the chance of occurrence of error is less when compared with the existing system.

**10.2 DISADVANTAGES:**

* Machine errors are unavoidable when occurred. (Hardware failure, network failure, others).
* Reach to geographically scattered student.
* Reducing time in activities
* Paperless admission with reduced man power
* Operational efficiency
* The predictions made are not 100% accurate but accurate to an acceptable value.

**11. CONCLUSION**

A model was developed to determine the admission of a student to the interested universities. The following parameters were taken into consideration:GRE Score,TOEFL Score,University Ranking,SOP,LOR,CGPA.From the validations, we can find out that the above parameters greatly contributed in determining the “Chance of Admit” into an university. Different models - Multiple Linear Regression, Random Forest Regression were taken into consideration.Out of the 2 models, Multiple Linear Regression output formed other models with a R2 score of 0.808.Hence Multiple Linear Regression was adopted in predicting the results.The project uses a Linear regressor to predict the output and a web application is built to make the UI more accessible and easy using various technologies such as python, React JS, HTML5, CSS, Flask, Scikit, Matplot, Numpy, Pandas, Seaborn and other libraries. After the deployment of the web application, it can be accessed from anywhere with internet connection. This project reduces the long hours of analysis to predict the eligibility of the admission to a rated university.

**12. FUTURE SCOPE**

The future scope of this project is very broad. Few of them are:

* This can be implemented in less time for proper admission process.
* This can be accessed anytime anywhere, since it is a web application provided only an internet connection.
* The user need not travel a long distance for the admission and his/her time is also saved as a result of this automated system.
* Develop a community consisting of faculty, alumni and aspirants to get to know about the university more
* A future update could have chat space where candidates, faculties, current students of the university and alumni can interact and candidates can get their doubts resolved instantly.
* Get in touch with grad-schools’ and professors and determine other important factors that play a key role in increasing the chances of admission.
* Alternatively, distributed big-data processing techniques could be explored if the no. of users using the website increases exponentially during the course of time.
* To deal with huge volumes of data in the future (Both - applicants and university details), cloud based storages (IBM cloud, AWS, GCP, AZURE) and NoSQL databases (MongoDB, Redis, etc.) could be used instead of the traditional RDBMS storage.

**13.Appendix**

**Source Code**

[index.html](https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Final%20Deliverables/Templates/index.html)

[chance.html](https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Final%20Deliverables/Templates/chance.html)

[nochance.html](https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Final%20Deliverables/Templates/nochance.html)

[finalcode.py](https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Final%20Deliverables/finalcode.py)

**GitHub Link :** <https://github.com/IBM-EPBL/IBM-Project-15550-1659600285>

**Website Link :** [University Admit Eligibility Predictor](http://universityeligibilitypredictor.pythonanywhere.com/)

**Project Demo Link :** [Demo.mp4](https://github.com/IBM-EPBL/IBM-Project-15550-1659600285/blob/main/Final%20Deliverables/Demo.mp4)