RCCIIT PLACEMENT DATA ANALYTICS

REPORT OF MINOR PROJECT SUBMITTED FOR PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF COMPUTER APPLICATION

By

MONODIPTA MAITY

Registration No- 231170510030 of 2023-2024

University Roll No- 11771023030

TIASHA HAIT

Registration No- 231170510063 of 2023-2024

University Roll No- 11771023063

VIVEK DEY

Registration No- 231170510066 of 2023-2024

University Roll No- 11771023066

SAHELI DEB

Registration No- 231170510080 of 2023-2024

University Roll No- 11771023081

UNDER THE SUPERVISION OF

DR. MANAS GHOSH
Assistant Professor, Department of Computer Application, RCC INSTITUTE OF INFORMATION TECHNOLOGY

AT
RCC INSTITUTE OF INFORMATION TECHNOLOGY
[Affiliated to West Bengal University of Technology]
CANAL SOUTH ROAD, BELIAGHATA, KOLKATA – 700 015

RCC INSTITUTE OF INFORMATION TECHNOLOGY

KOLKATA – 7OOO15, INDIA

CERTIFICATE

The report of the Project titled—'RCCIIT PLACEMENT DATA ANALYTICS' submitted by Monodipta Maity (Roll No.: 11771023030), Tiasha Hait (Roll No.: 11771023063), Vivek Dey (Roll No.: 11771023066), Saheli Deb (Roll No.: 11771023081) has been prepared under our supervision for the partial fulfillment of the requirements for MCA degree in Maulana Abul Kalam Azad University of Technology

The report is hereby forwarded.

Dr. Manas Ghosh
Department of Computer Application
RCCIIT, Kolkata

Countersigned by

……..
Biswanath Chakraborty
HoD, Department of Computer Application
RCC Institute of Information Technology, Kolkata – 700 015, India

RCC INSTITUTE OF INFORMATION TECHNOLOGY

KOLKATA – 7OOO15, INDIA

CERTIFICATE of ACCEPTANCE

The report of the Project titled—'RCCIIT PLACEMENT DATA ANALYTICS' submitted by Monodipta Maity (Roll No.: 11771023030), Tiasha Hait (Roll No.: 11771023063), Vivek Dey (Roll No.: 11771023066), Saheli Deb (Roll No.: 11771023081) is hereby recommended to be accepted for the partial fulfillment of the requirements for MCA degree in Maulana Abul Kalam Azad University of Technology

Name of the Examiner	Signature with Date
1. ……	…….
2. ……..	……...

ACKNOWLEDGEMENT

I express my sincere gratitude to Dr. Manas Ghosh Assistant Professor, Department of Computer Application, RCCIIT for extending his valuable times for us and guide us for this Project.

We are also indebted to Pramit Ghosh, Moumita Banerjee, Lidia Ghosh and Subhasis Chowdhury for their unconditional help and inspiration.

Last but not the least we would like to express our gratitude for the institution as a whole for providing a great learning platform wherein we can excel and grow as individuals.

Date: 18/01/2025

Monodipta Maity
Reg. No.: 231170510030
Roll No.: 11771023030
MCA SEM-3, RCCIIT

Tiasha Hait
Reg. No.: 231170510063
Roll No.: 11771023063
MCA SEM-3, RCCIIT

Vivek Dey
Reg. No.: 231170510066
Roll No.: 11771023066
MCA SEM-3, RCCIIT

Saheli Deb
Reg. No.: 231170510080
Roll No.: 11771023081
MCA SEM-3, RCCIIT

Table of Contents

Abstract—

The incorporation of data-driven decision-making has transformed industries, including education and placement analytics. This research describes a complete system for predicting students' placement potential using machine learning. The system is built using a React.js frontend, an AWS-powered backend, and the XGBoost algorithm, allowing for efficient data entry, accurate prediction, and informative result presentation. The user-friendly interface allows users to upload data, view interactive visualizations, and receive processed outputs. The backend assures scalability and computational efficiency, and the machine learning model is highly accurate. This paper describes the system's methodology, architecture, and performance, emphasizing its dependability and practical use.

I. INTRODUCTION

At the time of rapid technological changes and intense job competition, it is a necessity that the educational institutions and colleges should be ready to train their students according to industry requirements. Also, judging by the proportion of graduates obtaining employment, another powerful yardstick to assess the effectiveness of an educational programme is the calculation of the percentage of graduates getting work. Because of this, predicting whether students will find jobs after graduation has become very important for universities, colleges, and hiring companies. These predictions help schools improve students' skills, adjust their courses to match what employers want, and make hiring processes more effective. To tackle this challenge, there is a need for a system that uses data, can handle large amounts of information, and accurately predicts students' chances of getting a job based on their performance.

This initiative seeks to tackle this obstacle by harnessing advanced machine learning tools and cloud computing facilities to establish a dependable placement forecasting framework. The heart of this resolution is the use of XGBoost technique, an extremely powerful and broadly used artificial intelligence technique, optimized for the processing of structured, yet big data sets while providing superior accuracy and rate. Based on student inputting academic performance (10th and 12th exam, GPA) and their preselected academic stream and other demographic information (gender), the system will predict students' "Placeable" or "Unplaceable" probability. Enable insights to let organizations actively enhance student placements and help cover educational or vocational deficiencies.

The system integrates a React.js and Material-UI-based frontend with a cloud-hosted backend in the AWS Mumbai Region (ap-south-1). Keyword Use "rewrite" at the beginning. The front-end delivers an interactive interface that permits users to submit Excel-formatted files, procure comprehensive output files, and exhibit crucial insights via graphs and diagrams. A Spring Boot API for communication between the user and the server, and a Python Machine Learning tool that manipulates input data and creates forecasts. This architecture allows fast, natural user-to-system communication, which leads to both effectiveness and scalability.

One of the major advantages of this project is its capacity to combine predictive analytics with actionable outputs to maintain the trustworthiness and resilience of the model such that statistical inferential approaches, including t-tests and effectiveness measures such as cross-validated precision and area under the curve-based ROC analysis are essential. Consequently, the allocation of the user-interface within an Amazon S3 repository featuring general web accessibility facilitates promptly available usage, rendering it broadly utilizable across varied interested parties.

We employ cutting-edge ML methods, cloud services, and easy-to-use design concepts to create a complete system for figure location prediction. This meets the critical needs for prediction in the field of higher education, and it will provide a foundation for future developments of the field, including integrating richer and more complex data, implementing new computing approaches, and expanding predictive power to other areas such as career counselling, and employment planning. This novel blueprint marks a significant advance at the intersection of teaching, and invention, and questioning of data, that will have wide impact on educational and professional communities.

II. PROBLEM ANALYSIS

The main objective of this project is to build an automated, high throughput system for student placement prediction using student attributes via machine learning algorithms. Here is a brief problem analysis for this system:

Data Processing & Model Accuracy: How to process the student data in large scale and give accurate prediction. Input data has many attributes (e.g., academic performance, stream, gender, CGPA), and these need to be pre-processed, normalized and then fed into a trained XGBoost model to make the predictions.

Real-time File Handling: The system requires real-time file uploads and processing. Cloud Integration: Since the system is integrated with AWS services (EC2 for computations and S3 for frontend storage), efficient file transfers, system stability and robustness are important to prevent system failures.

Machine Learning Interpretation: The performance of the model should be clear and interpretable for the end-user to understand how different factors (like CGPA, stream, etc.) impact the prediction of placement status. Scalability & Reliability: With large number of input files and continuous model retraining, scalability and system reliability are major challenges.

III. TOOLS AND PLATFORMS

RCCIIT Placement Data Analytics is built using various tools, technologies and platforms. Following are the tools, technologies and platforms in brief:

A. Development Platform: Microsoft Windows11

B. Programming Languages: JavaScript, Java, Python

C. Styling Language: HTML, CSS

D. IDEs: VSCode, IntelliJ

E. Frontend:

1) Frontend Hosting platform: Amazon Web Services (AWS) - Simple Storage Service (S3)

2) Frameworks/Libraries: React.js, MaterialUI, Axios, Cookies.js

3) Build tool: Vite

F. Backend:

1) Backend deployment platform: Amazon Web Services (AWS) - Elastic Compute Cloud (EC2)

2) API Layer:

a) Frameworks/Libraries: Spring Core, Spring Boot, Spring Data JPA (with MySQL Connector)

b) Database: MySQL

c) Build tool: Maven

3) Analytics Layer:

a) Frameworks/Libraries: Scikit-learn

G. Utility Tools:

1) Version control system: git

2) Project repository hosting: GitHub.com

IV. LITERATURE REIVIEW

In recent years, the prediction of student placements using machine learning has received significant attention, with various approaches explored to enhance the accuracy and utility of such systems. Several studies have investigated the effectiveness of different algorithms and methodologies in predicting placement outcomes based on historical and behavioural data.

Varsha K. Harihar, in [1], with collaboration of D. G. Bhalke used multi-class classifier and used predictors that are more effective Naive Bayes (NB), Sequential Minimal Optimization (SMO), Multilayer Perceptron MLP, Logistic Model Tree LMT, Simple Logistic Regression, wherein their work to predict undergraduate placement results and assess using the respective performance metrics on these metrics the prediction was to measure the Accuracy, Precision, Recall, F-measure, and finally, ROC-AUC. The Naive Bayes classifier's prediction was based on Bayes' theorem:

$$P(A|B) = \frac{P(B|A)\,.\,P(A)}{P(B)}$$

where P(A|B) represents the posterior probability, P(B|A) is the likelihood, P(A) is the prior probability, and P(B) is the marginal probability. The study concluded that these classifiers could provide valuable insights into placement prediction [1].

Another study by Erman Çakıt and Metin Dağdeviren [2] focused on regression-based approaches to predict the percentage of student placements. Machine learning techniques such as linear regression, ridge regression, LASSO regression, elastic net regression, and XGBoost were employed. Among these, XGBoost demonstrated the highest predictive accuracy, evaluated using metrics such as Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Coefficient of Determination (R^2):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(P_i - A_i)^2}\,, \qquad M\,AE = \frac{1}{n}\sum_{i=1}^{n}|P_i - A_i|\,,$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(P_i - A_i)^2}{\sum_{i=1}^{n}A_i^2}$$

The results highlighted the suitability of regression models for predicting placement percentages [2]. K. Eswara Rao et al. [3] proposed an advanced career prediction model using XGBoost, incorporating both behavioural and academic attributes collected from students and alumni. The study compared XGBoost against classifiers such as Decision Trees, Random Forest, and Logistic Regression, demonstrating that XGBoost outperformed other models in terms of accuracy and precision. The XGBoost algorithm minimizes the following objective function:

$$L(t) = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t)}\right) + \sum_{k=1}^{k} \Omega(f_k)$$

where $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda||\omega||^2$, T is the number of leaves, and ω is the leaf weight. The research provided an in-depth evaluation of XGBoost's efficiency for student placement predictions [3].

In [4], Laxmi Shanker Maurya et al. explored the application of multiple classifiers, including SVM, Random Forest, Decision Trees, and Neural Networks, for predicting student placements in the IT industry. The study highlighted that Stochastic Gradient Descent (SGD) achieved the highest accuracy of 91.17%. The classification accuracy was calculated as:

where TP represents True Positive, TN is True Negative, FP is False Positive, and FN is False Negative. The use of confusion matrices and AUC-ROC curves further enhanced the evaluation process, making this research valuable for placement strategies [4].

Finally, Archana et al. [5] focused on a comparative analysis of Naive Bayes and K-Nearest Neighbours (KNN) for predicting placements. The system integrated diverse features such as academic records, technical and social skillsets, and behavioural information to enhance prediction accuracy. While Naive Bayes provided probabilistic insights, KNN was shown to outperform it in handling non-linear data patterns. The research workflow involved preprocessing historical student data, training models, and comparing accuracy, precision, and recall metrics. This system not only helped institutions refine their placement strategies but also provided students with actionable feedback [5].

M. S. Surya *et al*. [6] proposed a prediction model for estimating pupil's placement which outperforms standard classification algorithms.

In another study by Aravind *et al*. [7] tests the established machine learning models like regression (linear, light GBM and K-neighbor), XGBoost, decision tree, random tree etc. to predict the student's probability in getting placed.

In forecasting campus placement success for students, several algorithms have been studied by several authors [8][9][10].

In their research work, Tadla *et al*. emphasized the benefits of remote and flexible assessments, showing how technology can modernize the hiring process [11].

Manike *et al*. put forward XGBoost-based technique to evaluate whether the student will secure placement using numerous EDAs [12].

V. PROPOSED METHODOLOGY

This is implemented as a full stack application to ensure smooth experience of a user and the processing of backend to be robust

A. Frontend Design

The User Interface is build using react.js, it is a JavaScript library for building user interfaces and styling is done using Material -UI, it is a popular component library of react. Together these technologies ensure the Interactive Ness, user-friendly and appealing nature of the application. Let's discuss about the key feature of the frontend:

User authentication, restricting access to authorized users and maintaining data confidentiality is the key features of the Login page. Once the user is logged in, he/she is redirected to the Home page, this serves as the main feature of data processing. In the Home page, the user is required to upload the excel file with the required attributes containing the student data for analysis. After processing, the result is presented in various formats such as pie charts, and bar charts. The visualization helps us to quick understand the following trends and insights. Further, the application allows the user to download the predicted excel file named as "Analytics.xlsx". For further analysis. Amazon S3 bucket where the front end is hosted with public web accessed enabled. The setup enables use high accessibility, scalability and fast content delivery, making it accessible from anywhere.

B. Backend Work Flow

The backend is deployed on an Amazon EC2 instance which is running on a Windows operating system in the AWS Mumbai region (ap-south-1). File upload, executing the machine learning model and returning the processed results is totally handled by the backend. It consists of two main components:

The API Endpoint Program, developed using Spring Boot, it acts as the intermediary between the front end and the machine learning model. Firstly, on receiving an input file, it saves the file to a designated input/ folder on the EC2 instance and monitors the for the file in the output/ folder. The API Program is structured in such a way that it can pull the file from the output folder 30 times keeping the client connectivity alive. Once the processed file is available in the output/ folder, it is sent back to the client for further processing and visualization.

The Machine learning Processing Program, which is developed in Python, continuously monitors the input/ folder for new files. Upon detecting a file in the input/ folder, the program then loads the machine learning model which is previously trained and then process the data and generates a file in the output folder named as "Analytics.xlsx". The output file contains the prediction about the student's placement.

C. File Handling Work Flow

The file handling work flow ensures seamless communication between the frontend backend and the machine learning model:

1)Input folder: It basically receives the input file which is uploaded as excel file from the frontend processing.

2)Output folder: It is used to store the output excel file, which is then retrieved by the API endpoint program and send back to the client

D. Machine Learning Model Algorithm: XGBoost

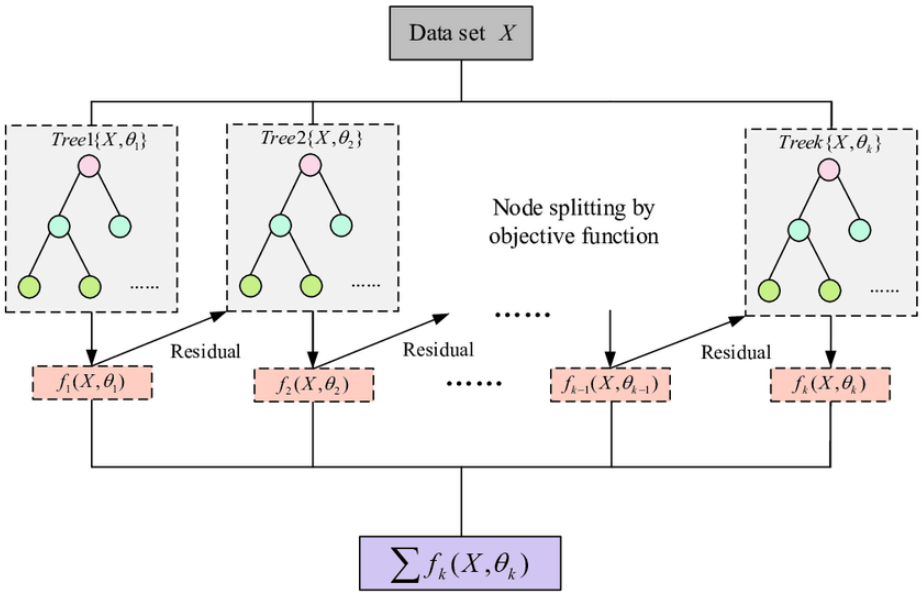


Figure: Diagram of XGBoost Algorithm

Algorithm of XGBoost:

1) Initialization

- Input: Dataset with features (XXX) and target labels (yyy).
- Start with an initial prediction, often the mean of the target values for regression or log odds for classification.

2) Iterative Boosting Process

For each boosting round:

Step 2.1: Compute Gradients and Hessians

- Gradients represent the errors (residuals) between the predicted and actual values.
- Hessians represent the second-order approximation of the loss function, helping to refine updates.

Step 2.2: Fit a Decision Tree

- A decision tree is built to minimize the loss.
- Splits are chosen to maximize the reduction in loss, considering both gradients and Hessians.

Step 2.3: Optimize Leaf Weights

- Each leaf in the decision tree is assigned a weight that minimizes the loss for the samples it contains.

Step 2.4: Update Model Predictions

- The predictions are updated by adding the contributions from the current tree, scaled by the learning rate.

3) Final Prediction

After running the specified number of boosting rounds, combine the contributions from all the trees to make the final prediction.

XGBoost(Extreme Gradient Boosting), it is one of the powerful machine learning algorithm that expertise's in structured data classification tasks. It mainly uses gradient-boosting framework to iteratively improve the prediction accuracy by the minimizing the errors. The key advantage of XGBoost include:

1) Efficiency: It is mainly suitable for large-scale datasets as it is highly optimized for speed and performance.

2) Robustness: It can handle properly the missing values and also the categorial variables effectively, ensuring reliable predictions in real-world senarios.

3) Accuracy: Its main learning approach combines multiple decision trees to enhance predictive performance, reducing the overfitting.

E. Attributes Used

The following features from the dataset are used in the training of the machine learning model:

1)Input Features:

a) DOB: This date of birth of the student, it is used to calculate age, which may correlate with the placement documents.

b) Gender: It was encoded as categorical values which mainly account for potential gender-based trends in placement.

c) 10th_marks and 12th_marks: Academic performance in secondary (10) and higher secondary (12) examinations, serving as indicators of consistency and diligence.

d) Stream: It is the academic discipline, such as CSE, IT, ECE, EE or AEIE, reflecting the domain expertise of the student.

e) CGPA: The cumulative grade point average, indicating overall academic performance.

2)Target Variable:

a) PlacedOrNot: It is basically a binary classification indicating whether a student is "Placed" or "Unplaced".

3)Prediction Output:

a) Placebility: It is the predicted classification indicating whether a student is "Placeable" or "Unplaceable".

F. Model Evaluation

The evaluation of the model was done using statistical and machine learning metrics to ensure its reliability:

1)T-Test:

a) It is a statistical test to compare the model's mean accuracy against a baseline (50%).

b) Results: T-statistic: 48.0787

P-value: < 0.05, it indicates the statistically significant difference from the baseline.

2)Cross-Validation:

a) 50-fold cross-validation across multiple data splits was used to evaluate the model, ensuring generalizability.

b) Metrics: Mean Accuracy: 79.68%

Standard Deviation: 4.32%, which reflects consistent performance across folds.

3)ROC-AUC Metrics:

a) Micro-Average AUC-ROC: 0.95, capturing the overall predictive power across all classes.

b) Macro-Average AUC-ROC: 0.93, reflecting balanced performance across individual classes.

G. SMOTE for Class Imbalance

SMOTE (Synthetic Minority Oversampling Technique) was applied to address class imbalance in the dataset. For underrepresented classes SMOTE generates synthetic samples. Post-SMOTE, the class distribution was equalized, ensuring the model learned from a representative dataset.

H. Final Model Performance

GridSearchCV identified the optimal configuration for the XGBoost model using the Hyperparameter tuning:

a) Learning Rate: 0.01

b) Max Depth: 5

c) Number of Estimators: 200

The final accuracy of the model achieved was 81.21% on the test set, which demonstrate its robustness and reliability. The overall classification report revealed:

a) Precision and Recall: High metrics for the "Placeable" class, which indicates the model's effectiveness in identifying placement potential.

b) F1-Score" Balanced metrics for both "Placeable" and "Unplaceable" classes.

I. Data Visualization

The processed data is visualized in the frontend to provide actionable insights:

1) Pie charts: It displays the proportion of student classified as "Placeable" and "Unplaceable". This helps the client to understand the overall placement trends

2)Bar charts: It basically highlights relationships between key attributes (e.g., CGPA, stream, and placement potential), enabling deeper analysis and strategic decision-making.

This comprehensive methodology integrates advanced machine learning techniques, cloud-based architecture, and intuitive user interfaces to deliver a powerful solution for placement prediction and analysis.

VI. SCREENS OVERVIEW

This section describes the user interface and dataset-related screens that are integral to the RCCIIT Placement Data Analytics (PDA) application. All screens have very important task in providing a consistent data input, analysis, forecasting, and visualization pipeline:

	A	B	C	D	E	F	G	H
1	Sl No.	DOB	Gender	10th_marks	12th_marks	Stream	CGPA	PlacedOrNot
2	1	13-05-1998	MALE	91	76.6	CSE	7.33	Placed
3	2	03-09-1999	MALE	86.14	88.6	CSE	8.31	Placed
4	3	30-08-1997	MALE	84.8	76.6	CSE	7.86	Placed
5	4	23-09-1999	MALE	89.3	78	CSE	8.2	Placed
6	5	25-11-1998	MALE	77.8	73.5	CSE	7.16	Placed
7	6	07-02-2000	MALE	89.3	90.6	CSE	8.62	Placed
8	7	13-12-1998	MALE	96	97.5	CSE	9.22	Placed
9	8	13-12-1998	MALE	94.2	96.25	CSE	8.99	Placed
10	9	30-12-1998	MALE	92.8	82	CSE	8.75	Placed
11	10	30-07-1998	MALE	91	91.75	CSE	7.92	Placed
12	11	06-04-2000	MALE	87.57	87	CSE	8.02	Placed
13	12	19-08-1998	MALE	93.5	82	CSE	7.08	Placed
14	13	06-11-1999	FEMALE	95	75.4	CSE	8.15	Placed
15	14	21-10-1998	MALE	89.14	76.6	CSE	9.01	Placed
16	15	24-05-1999	FEMALE	93.14	83.4	CSE	8.34	Placed
17	16	20-10-1999	MALE	95	74.8	CSE	8.26	Placed
18	17	27-05-1999	FEMALE	77.71	76.8	CSE	8.53	Placed
19	18	22-04-1999	MALE	89.42	81.4	CSE	8.89	Placed
20	19	29-05-1998	FEMALE	97.4	83.4	CSE	8.6	Placed
21	20	09-08-2000	MALE	83.6	68.6	CSE	8.45	Placed
22	21	26-10-1998	MALE	92.4	94	CSE	8.46	Placed
23	22	26-05-1998	FEMALE	95.2	96.5	CSE	9.31	Placed
24	23	25-09-1999	FEMALE	90.57	76.6	CSE	8.69	Placed
25	24	16-03-2000	FEMALE	89.8	90.2	CSE	8.47	Placed
26	25	28-01-1999	MALE	95.8	96.25	CSE	8.42	Placed
27	26	06-11-1997	MALE	84.7	81.4	CSE	7.35	Placed
28	27	07-09-1998	MALE	81.7	74.4	CSE	7.8	Placed
29	28	06-03-1998	MALE	96	93	CSE	9.39	Placed

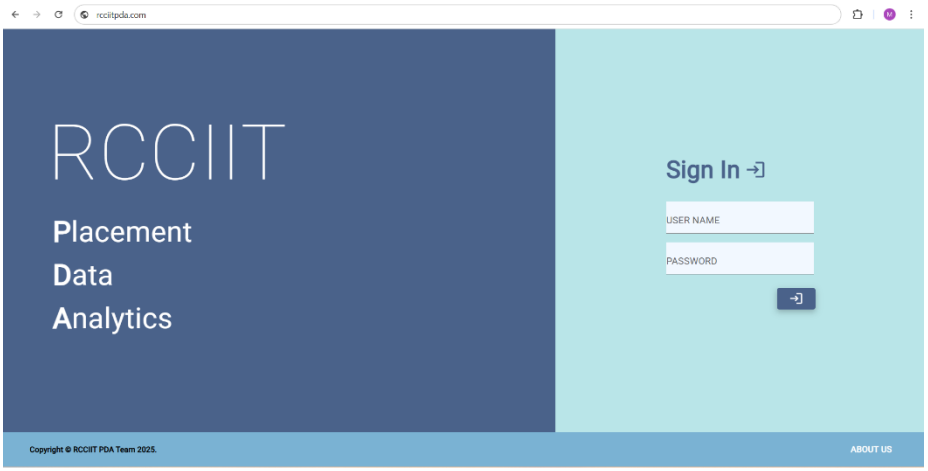
TrainingDataSetRCCIIT

Training Dataset: A detailed dataset containing information about students, including academic records (10th and 12th board marks), specialization, CGPA, and placement status.

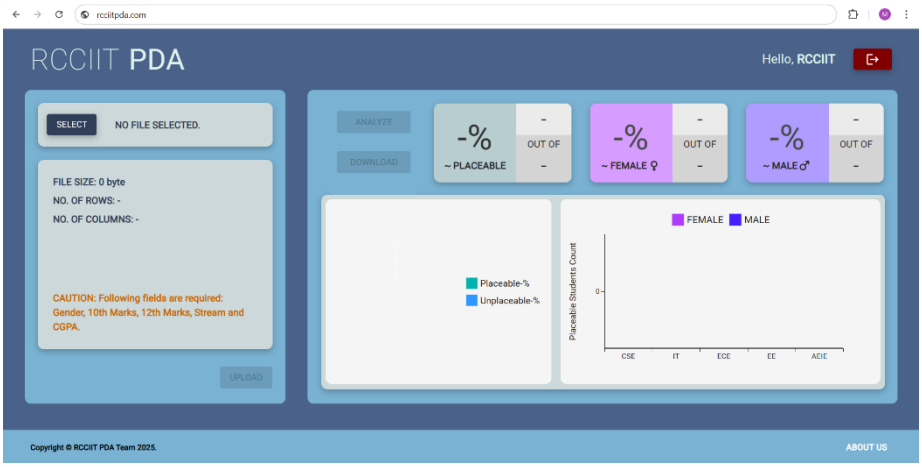
	A	B	C	D	E	F	G
1	Sl No.	DOB	Gender	10th_marks	12th_marks	Stream	CGPA
2	1	05-01-1998	MALE	88.3	85.5	CSE	8.12
3	2	15-07-1999	FEMALE	91.2	79.4	IT	7.89
4	3	28-03-1997	MALE	79.8	85.6	EE	7.47
5	4	12-02-1998	FEMALE	85.2	90.3	ECE	8.24
6	5	22-11-1997	MALE	78.4	82.7	AEIE	6.98
7	6	03-08-1999	FEMALE	92.5	88.1	CSE	8.55
8	7	19-05-1998	MALE	84.6	91.3	IT	7.98
9	8	26-01-1998	FEMALE	90.2	79.9	EE	7.34
10	9	17-09-1998	MALE	86.4	83.2	ECE	8.12
11	10	03-04-1997	FEMALE	89.1	85.3	AEIE	7.99
12	11	25-12-1998	MALE	83.7	91	CSE	8.33
13	12	05-06-1999	FEMALE	79.2	80.3	IT	7.65
14	13	09-08-1997	MALE	94.1	77.4	EE	7.51
15	14	10-10-1999	FEMALE	80.5	89.1	ECE	8.09
16	15	01-02-1998	MALE	87.3	78.8	AEIE	7.47
17	16	13-11-1997	FEMALE	92.4	90.2	CSE	8.51
18	17	30-07-1999	MALE	85.7	77.1	IT	7.99
19	18	22-01-1998	FEMALE	91.5	86.8	EE	8.33
20	19	17-10-1998	MALE	89.9	79.3	ECE	8.06
21	20	01-04-1997	FEMALE	84.3	87.5	AEIE	7.75
22	21	14-03-1999	MALE	90	91.8	CSE	8.42
23	22	30-12-1998	FEMALE	80.6	84.7	IT	7.59
24	23	06-02-1997	MALE	94.3	88.9	EE	8.16
25	24	11-08-1998	FEMALE	87.5	91.4	ECE	8.39
26	25	17-05-1999	MALE	88.1	79.7	AEIE	7.52
27	26	20-01-1997	FEMALE	79.5	80.3	CSE	7.23
28	27	28-10-1998	MALE	81.4	85.6	IT	7.87
29	28	22-09-1999	FEMALE	85	84.2	EE	7.92

Generated_Test_Data

Testing Dataset: A synthetic dataset showcasing test records of students, with fields such as marks, stream, CGPA, and gender, for evaluating the prediction model.



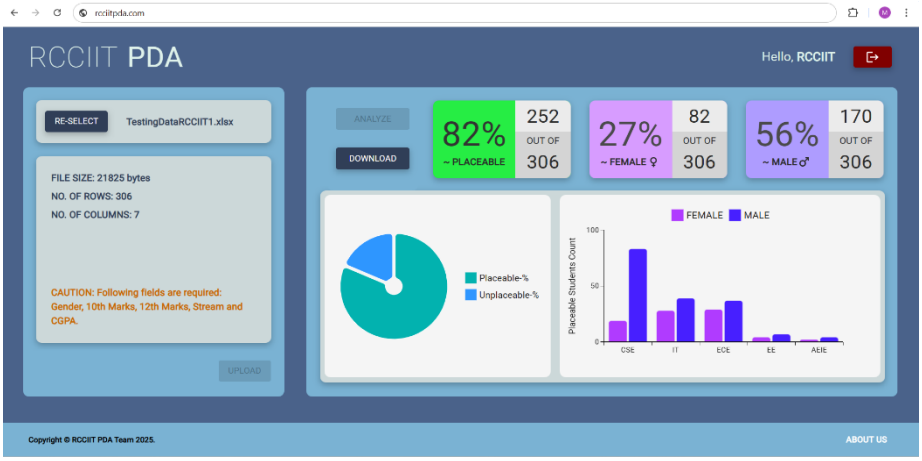
Login Screen: User authentication interface for accessing the RCCIIT Placement Data Analytics application.



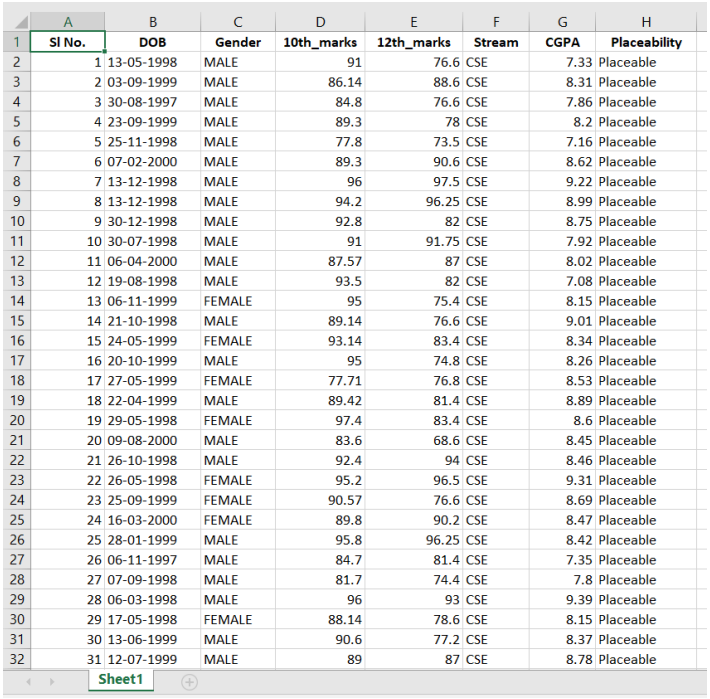
Home Screen: Default state of the dashboard when no file is uploaded, awaiting user interaction

.

Data Input Screen: Users can upload the dataset for placement analysis, showing file details like size, number of rows, and required fields.



Final Output Screen: Displays analysed results, including placement statistics, gender-wise breakdown, and department-wise placement insights.



Sl No.	DOB	Gender	10th_marks	12th_marks	Stream	CGPA	Placeability
1	13-05-1998	MALE	91	76.6	CSE	7.33	Placeable
2	03-09-1999	MALE	86.14	88.6	CSE	8.31	Placeable
3	30-08-1997	MALE	84.8	76.6	CSE	7.86	Placeable
4	23-09-1999	MALE	89.3	78	CSE	8.2	Placeable
5	25-11-1998	MALE	77.8	73.5	CSE	7.16	Placeable
6	07-02-2000	MALE	89.3	90.6	CSE	8.62	Placeable
7	13-12-1998	MALE	96	97.5	CSE	9.22	Placeable
8	13-12-1998	MALE	94.2	96.25	CSE	8.99	Placeable
9	30-12-1998	MALE	92.8	82	CSE	8.75	Placeable
10	30-07-1998	MALE	91	91.75	CSE	7.92	Placeable
11	06-04-2000	MALE	87.57	87	CSE	8.02	Placeable
12	19-08-1998	MALE	93.5	82	CSE	7.08	Placeable
13	06-11-1999	FEMALE	95	75.4	CSE	8.15	Placeable
14	21-10-1998	MALE	89.14	76.6	CSE	9.01	Placeable
15	24-05-1999	FEMALE	93.14	83.4	CSE	8.34	Placeable
16	20-10-1999	MALE	95	74.8	CSE	8.26	Placeable
17	27-05-1999	FEMALE	77.71	76.8	CSE	8.53	Placeable
18	22-04-1999	MALE	89.42	81.4	CSE	8.89	Placeable
19	29-05-1998	FEMALE	97.4	83.4	CSE	8.6	Placeable
20	09-08-2000	MALE	83.6	68.6	CSE	8.45	Placeable
21	26-10-1998	MALE	92.4	94	CSE	8.46	Placeable
22	26-05-1998	FEMALE	95.2	96.5	CSE	9.31	Placeable
23	25-09-1999	FEMALE	90.57	76.6	CSE	8.69	Placeable
24	16-03-2000	FEMALE	89.8	90.2	CSE	8.47	Placeable
25	28-01-1999	MALE	95.8	96.25	CSE	8.42	Placeable
26	06-11-1997	MALE	84.7	81.4	CSE	7.35	Placeable
27	07-09-1998	MALE	81.7	74.4	CSE	7.8	Placeable
28	06-03-1998	MALE	96	93	CSE	9.39	Placeable
29	17-05-1998	FEMALE	88.14	78.6	CSE	8.15	Placeable
30	13-06-1999	MALE	90.6	77.2	CSE	8.37	Placeable
31	12-07-1999	MALE	89	87	CSE	8.78	Placeable

Output Dataset: Sample dataset showing student records with fields like DOB, Gender, Marks, Stream, CGPA, and Placeability status.

VII. CONCLUSION AND FUTURE SCOPE

A. Conclusion:

Joining machine-learning solutions in a cloud framework can significantly push the boundaries of employment prediction, in terms of technology, while this effort requires as one of the highest performing team learning algorithms (XGBoost) to be employed for producing highly accurate, tough, and capable-of-maturation forecasts. The system links a React front-end, an AWS-hosted Spring Boot API, and Python ML processing making it easy to use and quick to work. By putting the system in AWS's Mumbai area, it becomes more trustworthy and easier to reach showing a new cloud-first way of building apps.

The project breaks new ground by merging raw data analysis predictive modelling, and easy-to-use visualization into one unified system. The model's performance has proven to be meaningful and dependable backed up by cross-validation, t-tests, and AUC-ROC measurements. What's more, the frontend's use of pie charts and bar graphs to show data helps stakeholders make better choices.

B. Future scope:

Any changes at this juncture would include investigatory focus on more sophisticated deep learning architectures like networks of neurons, to comprehend patterns which are more complex. More variables, such as employment history before signing on with the company, internship attendance, or personality testing, could probably boost the accuracy of the portrayal. Team-ups with learning management systems (LMS) and universities can offer more detailed data and help this tool reach more people. Joining forces with platforms like Coursera or LinkedIn could back up claims about students' job readiness. The front end could show live dashboards with heat maps and AI suggestions to help students boost their skills based on what the system learns. Also, making the website or app work in different languages will help more people use it easily.

VIII. RESULTS

This section presents the results of the machine learning version used to predict placement outcomes. Statistical metrics Statistical analysis and visualization is included to check version performance.

A. Model Performance

The model illustrates the robust performance of the final analysis during the testing process, and when introducing cross-validation.

1) Cross-Validation: Accuracy scores from 50 cross-validations are summarized in Fig. 1.

- Mean accuracy: 79.68%
- Standard deviation: 4.32%

$$Mean\ Accuracy = \frac{1}{n}\sum_{\{i=1\}}^{n} Accuracy_i$$

Where n = 50 folds.

2) Final Test Set Performance:
- Accuracy: 81.21%
- Precision, recall, and F1-scores for each class are reported in Table 1.

Table 1: Classification Report

Class	Precision	Recall	F1-Score	Support
0 (Unplaceable)	0.71	0.75	0.73	55
1 (Placeable)	0.73	0.69	0.71	55
2 (High Confidence)	1.00	1.00	1.00	55
Overall Accuracy	-	-	81.21%	165

B. Statistical Analysis of Accuracy

The statistical significance of mean accuracy from cross-validation compared to 50% baseline accuracy was assessed using the Welch-t test. The results are as follows:

-t-Statistic:48.08
- p-Value:0.0000

The results are statistically significant ($p < 0.05$), indicating that the model's accuracy is significantly higher than the baseline. Fig. 1 illustrates the histogram of accuracy scores across the folds.

C. t-Test Analysis of Numerical Features

Welch's t-tests were also performed to evaluate the significance of numerical features (10th_marks, 12th_marks, and CGPA) in distinguishing between placement classes. The results are summarized in Table 2.

Table 2: t-text results for numerical features.

Feature	p-Value
10th Marks	1.00

12th Marks	1.00
CGPA	1.00

The p-values for all features exceed the threshold of 0.05, indicating no statistically significant difference between the groups. Fig. 2 shows the bar plot of t-test p-values for these features.

D. ROC Curve Analysis

The Receiver Operating Characteristic (ROC) curve was plotted to evaluate the model's performance in distinguishing between placement classes:

- Micro-average AUC-ROC: 0.95
- Macro-average AUC-ROC: 0.93

Fig. 3 illustrates the micro-average ROC curve, with an AUC of 0.95, confirming the model's excellent predictive ability.

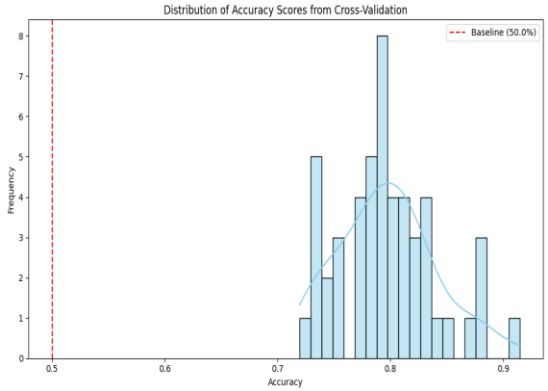


Figure 1: Distribution of accuracy scores from cross-validation folds. (Placement: Section IV-B, "Statistical Analysis of Accuracy.")

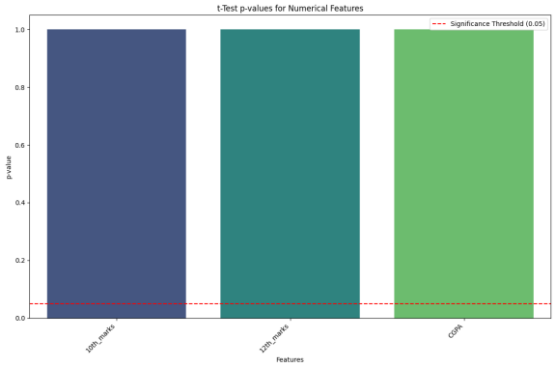


Figure 2: Bar plot of t-test p-values for numerical features. (Placement: Section IV-C, "t-Test Analysis of Numerical Features.")

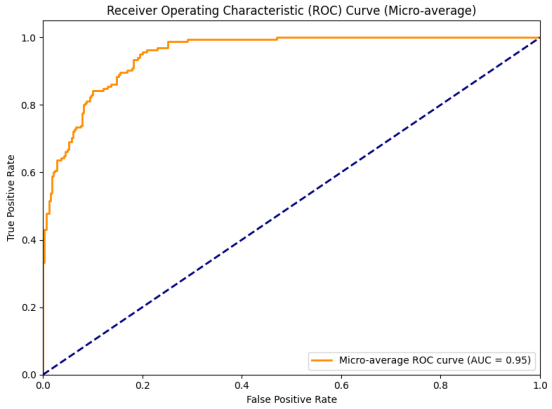


Figure 3: Micro-average ROC curve. (Placement: Section IV-D, "ROC Curve Analysis.")

REFERENCE:

[1] V. K. Harihar and D. G. Bhalke, "Student Placement Prediction System Using Machine Learning," SAMRIDDHI: A Journal of Physical Sciences, Engineering and Technology, 2020. DOI: 10.18090/samriddhi.v12iS2.17.

[2] 2. E. Çakıt and M. Dağdeviren, "Predicting the Percentage of Student Placement: A Comparative Study of Machine Learning Algorithms," Education and Information Technologies, 2021. DOI: 10.1007/s10639-021-10655-4.

[3] 3. K. E. Rao, B. M. Pydi, T. P. Vital et al., "An Advanced Machine Learning Approach for Student Placement Prediction and Analysis," International Journal of Performability Engineering, 2023. DOI: 10.23940/ijpe.23.08.p6.536546.

[4] 4. L. S. Maurya, M. S. Hussain, and S. Singh, "Developing Classifiers Through Machine Learning Algorithms for Student Placement Prediction," Applied Artificial Intelligence, 2021.

[5] 5. P. Archana, D. Pravallika, P. S. Priya, S. S. Priya, and S. Amitha, "Student Placement Prediction Using Machine Learning," Journal of Survey in Fisheries Sciences, vol. 10, no. 1, 2023.

[6] M. S. Surya, M. S. Kumar and D. Gandhimathi, "Student Placement Prediction Using Supervised Machine Learning," *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, Greater Noida, India, 2022, pp. 1352-1355.

[7] T. Aravind, B. S. Reddy, S. Avinash and J. G., "A Comparative Study on Machine Learning Algorithms for Predicting the Placement Information of Under Graduate Students," *2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India, 2019, pp. 542-546.

[8] Agrawal, V. S. ., & Kadam, S. S. . (2024). Predictive Analysis of Campus Placement of Student Using Machine Learning Algorithms. *Journal of IoT and Machine Learning* , *1*(2), 13–18.

[9] R. Kadu, P. J. Assudani, T. Mukewar, J. Kapgate and R. Bijekar, "Student Placement Prediction and Skill Recommendation System using Machine Learning Algorithms," *2024 International Conference on Inventive Computation Technologies (ICICT)*, Lalitpur, Nepal, 2024, pp. 401-408.

[10] Sajwan, S., Bhardwaj, R., Mishra, R., Jaiswal, S. (2024). Student Placement Prediction Using Machine Learning Algorithms. In: Deka, J.K., Robi, P.S., Sharma, B. (eds) Emerging Technology for Sustainable Development. EGTET 2022. Lecture Notes in Electrical Engineering, vol 1061. Springer, Singapore.

[11] V. S. Tadla, P. M. Singh, K. M. Thakkar and R. Adatkar, "Campus Placement using Machine Learning: An Extensive Review and Comparative Study of Machine Learning Methods," *2023 6th International Conference on Advances in Science and Technology (ICAST)*, Mumbai, India, 2023, pp. 427-430,

[12] M. Manike, P. Singh, P. S. Madala, S. A. Varghese and S. Sumalatha, "Student Placement Chance Prediction Model using Machine Learning Techniques," *2021 5th Conference on Information and Communication Technology (CICT)*, Kurnool, India, 2021, pp. 1-5.

APPENDIX

A. Code

1) App.jsx

```
import Signin from './pages/Signin';
import Home from './pages/Home';
import Aboutus from './pages/AboutUs';
import { Route, Routes } from 'react-router-dom';
function App() {
 return (
    <Routes>
       <Route path="/" element={<Signin />} />
       <Route path="/signin" element={<Signin />} />
       <Route path="/home" element={<Home />} />
       <Route path="/aboutus" element={<Aboutus />} />
    </Routes>
 );
}
export default App;
```

2) index.css

```
html, body {
   margin: 0;
   padding: 0;
   width: 100%;
   height: 100%;
}
```

3) main.jsx

```
import { StrictMode } from 'react'
import { createRoot } from 'react-dom/client'
import './index.css'
```

```
import App from './App.jsx'
import { BrowserRouter } from 'react-router-dom'

createRoot(document.getElementById('root')).render(
  <StrictMode>
     <BrowserRouter>
        <App />
     </BrowserRouter>
  </StrictMode>,
)
```

4) Signin.jsx

```
import {
   Backdrop,
   Box,
   Button,
   CircularProgress,
   Snackbar,
   TextField,
   Typography
} from "@mui/material";
import {useEffect, useState} from "react";
import Cookies from 'js-cookie';
import {useNavigate} from 'react-router-dom';
import axios from "axios";
import LoginIcon from '@mui/icons-material/Login';

function Signin() {

   const navigate = useNavigate();
   const [signinError, setSigninError] = useState(false);
   const [signinErrorMessage, setSigninErrorMessage] = useState("");
```

```
const [username, setUsername] = useState("");
const [password, setPassword] = useState("");
const [openBackdrop, setOpenBackdrop] = useState(false);
const [openSnackbar, setOpenSnackbar] = useState(false);
const [snackbarMessage, setSnackbarMessage] = useState("");
const [tokenValue, setTokenValue] = useState("");
const [nameValue, setNameValue] = useState("");
const [signinState, setSigninState] = useState(false);

function validate(username, password){
   if(username === "" && password === ""){
      setSigninErrorMessage("Fields are empty!");
      setSigninError(true);
      return false;
   }else if(username === ""){
      setSigninErrorMessage("Please enter the user name.");
      setSigninError(true);
      return false;
   }else if(password === ""){
      setSigninErrorMessage("Please enter the password.");
      setSigninError(true);
      return false;
   }
   return true;
}

function handleSignin() {
   if(validate(username, password)){
      setSigninErrorMessage("");
      setSigninError(false);
      setOpenBackdrop(true);
      axios.post('http://endpoint/v1/auth/sign_in', {
```

```
            password: password,
            name: username
        })
            .then(function (response) {
                setTokenValue(response.data.token);
                setNameValue(response.data.username);
                setOpenBackdrop(false);
                setSigninState(true);
                setSnackbarMessage("Sign In Successful!");
                setOpenSnackbar(true);
            })
            .catch((error) => {
                if (error.response) {
                    if (error.response.status === 406) {
                        setSigninErrorMessage("Invalid credentials. Please try again.");
                        setSigninError(true);
                    } else if (error.response.status === 404) {
                        setSigninErrorMessage("User not found! Please try again.");
                        setSigninError(true);
                    } else if (error.response.status === 500) {
                        setSnackbarMessage("Server error. Please try again later.");
                        setOpenSnackbar(true);
                    }
                } else {
                    setSnackbarMessage("Network error. Please check your connection.");
                    setOpenSnackbar(true);
                }
                setOpenBackdrop(false);
            })
    }
}
```

```
function handleSnackbarClose() {
    if(signinState){
        Cookies.set('token', tokenValue, { expires: 1 });
        Cookies.set('name', nameValue, { expires: 1 });
    }
    setOpenSnackbar(false);
}

function handleAboutusButton(){
    navigate('/AboutUs');
}

useEffect(() => {
    if(Cookies.get('token') !== undefined){
      navigate('/Home');
    }
});

return (
    <>
        <Box sx={{
                display: 'flex',
                width: '100vw',
                height: '92vh',
        }}>
            {(<Backdrop
                sx={(theme) => ({ color: '#fff', zIndex: theme.zIndex.drawer + 1 })}
                open={openBackdrop}
            >
                <CircularProgress color="inherit" />
            </Backdrop>)}
            <Box sx={{ width: '60vw',
```

```
        height: '92vh',
        display: 'flex',
        justifyContent: 'flex-start',
        backgroundColor: '#4A628A'
    }}>
      <Box
        sx={{ height: '92vh',
        display: 'flex',
        flexDirection: 'column',
        justifyContent: 'flex-start',
        backgroundColor: '#4A628A'
      }}>
        <Typography sx={{
          fontSize: '8em',
          fontWeight: 100,
          color: '#ffffff',
          marginTop: '15vh',
          marginLeft: '5vw'
          }}>
          RCCIIT
        </Typography>
        <Typography sx={{
          fontSize: '3em',
          fontWeight: 400,
          color: '#ffffff',
          marginLeft: '5.5vw',
          }}>
          <span style={{ fontWeight: 'bold' }}>P</span>lacement
        </Typography>
        <Typography sx={{
          fontSize: '3em',
          fontWeight: 400,
```

```
            color: '#ffffff',
            marginLeft: '5.5vw'
            }}>
            <span style={{ fontWeight: 'bold' }}>D</span>ata
          </Typography>
          <Typography sx={{
            fontSize: '3em',
            fontWeight: 400,
            color: '#ffffff',
            marginLeft: '5.5vw'
            }}>
            <span style={{ fontWeight: 'bold' }}>A</span>nalytics
          </Typography>
        </Box>
      </Box>
      <Box sx={{ width: '40vw',
        height: '92vh',
        backgroundColor: '#B9E5E8'
      }}>
        <Box sx={{
          display: 'flex',
          flexDirection: 'row',
          justifyContent: 'flex-start',
          marginTop: '28vh',
          marginLeft: '12vw',
          marginBottom: '1vw',
          alignItems: 'center'
        }}>
          <Typography sx={{
            fontSize: '2.5em',
            fontWeight: 900,
            color: '#4A628A',
```

```
    }}>
      Sign In
    </Typography>
    <LoginIcon fontSize="large" sx={{
      paddingLeft: '0.5vw',
      color: '#4A628A',
    }}/>
  </Box>
  {signinError && (<Typography sx={{
    fontSize: '1em',
    marginLeft: '12vw',
    fontWeight: 400,
    color: '#ffffff',
    backgroundColor: '#d40000',
    width: '38%',
    padding: '1%'
  }}>
    {signinErrorMessage}
  </Typography>)}
  <TextField
    id="user-name"
    label="USER NAME"
    variant="standard"
    type="text"
    color='#0d0047'
    sx={{
      marginTop: '1vh',
      marginLeft: '12vw',
      width: '40%',
      backgroundColor: '#f2f8ff',
      input: {
        color: 'black',
```

```
            fontSize: '20px',
            fontWeight: 'bold'
        }
    }}
    onChange={(e) => setUsername(e.target.value)}
/>
<TextField
    id="password"
    label="PASSWORD"
    variant="standard"
    type="password"
    color='#0d0047'
    sx={{
        marginTop: '2vh',
        marginLeft: '12vw',
        width: '40%',
        backgroundColor: '#f2f8ff',
        input: {
            color: 'black',
            letterSpacing: '3px',
            fontSize: '20px',
            WebkitTextSecurity: 'disc',
            MozTextSecurity: 'disc',
            textSecurity: 'disc'
        }
    }}
    onChange={(e) => setPassword(e.target.value)}
/>
<Button
    variant="contained"
    onClick={handleSignin}
    sx={{
```

```
                marginTop: '3vh',
                marginLeft: '24vw',
                backgroundColor: '#4A628A',
                boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)',
                '&:hover': {
                    backgroundColor: '#7AB2D3',
                }
            }}>
            <LoginIcon/>
        </Button>
        <Snackbar
            anchorOrigin={{ vertical: 'top', horizontal: 'right' }}
            open={openSnackbar}
            onClose={handleSnackbarClose}
            autoHideDuration={2000}
            message={snackbarMessage}
        />
    </Box>
</Box>
<Box sx={{
    display: 'flex',
    flexDirection: 'row',
    width: '100vw',
    height: '8vh',
    backgroundColor: '#7AB2D3',
    alignItems: 'center'
}}>
    <Typography sx={{
        fontSize: '0.8em',
        fontWeight: '900',
        marginLeft: '3vw'
    }}>
```

```
                {'Copyright © RCCIIT PDA Team '}
                {new Date().getFullYear()}
                {'.'}
              </Typography>
              <Box sx={{
                display: 'flex',
                flexGrow: 1,
                justifyContent: 'flex-end'
              }}>
                <Button
                  onClick={handleAboutusButton}
                  sx={{
                    marginRight: '3vw',
                    color: '#ffffff',
                    backgroundColor: '#7AB2D3',
                    fontWeight: '900'
                  }}
                >
                  About Us
                </Button>
              </Box>
            </Box>
          </>
        );
      }

      export default Signin;
```

5) Home.jsx

```
      import {Backdrop, Box, Button, CircularProgress, Snackbar, Tooltip, Typography} from
      "@mui/material";
      import {useEffect, useState} from "react";
```

```
import Cookies from "js-cookie";
import {useNavigate} from "react-router-dom";
import axios from "axios";
import LogoutIcon from '@mui/icons-material/Logout';
import * as XLSX from "xlsx";
import FemaleIcon from '@mui/icons-material/Female';
import MaleIcon from '@mui/icons-material/Male';
import { PieChart } from '@mui/x-charts/PieChart';
import {axisClasses, BarChart} from "@mui/x-charts";

function Home() {

    const navigate = useNavigate();
    const [fileName, setFileName] = useState("NO FILE SELECTED.")
    const [username, setUsername] = useState("");
    const [signoutState, setSignoutState] = useState(false);
    const [openBackdrop, setOpenBackdrop] = useState(false);
    const [openSnackbar, setOpenSnackbar] = useState(false);
    const [snackbarMessage, setSnackbarMessage] = useState("");
    const [selectButtonText, setSelectButtonText] = useState("Select");
    const [fileSize, setFileSize] = useState("0 byte");
    const [numberOfRows, setNumberOfRows] = useState("-");
    const [numberOfColumns, setNumberOfColumns] = useState("-");
    const [selectDisabled, setSelectDisabled] = useState(false);
    const [uploadDisabled, setUploadDisabled] = useState(true);
    const [analyzeDisabled, setAnalyzeDisabled] = useState(true);
    const [downloadDisabled, setDownloadDisabled] = useState(true);
    const [uploadError, setUploadError] = useState(false);
    const [uploadErrorMessage, setUploadErrorMessage] = useState("");
    const [selectFile, setSelectFile] = useState(null);
    const [analyzedFile, setAnalyzedFile] = useState(null);
    const [placeableCount, setPlaceableCount] = useState("-");
```

```
const [placeablePercentage, setPlaceablePercentage] = useState("-");
const [placeableColor, setPlaceableColor] = useState("");
const [femaleCount, setFemaleCount] = useState("-");
const [femalePercentage, setFemalePercentage] = useState("-");
const [maleCount, setMaleCount] = useState("-");
const [malePercentage, setMalePercentage] = useState("-");
const [CSECounts, setCSECounts] = useState({ MALE: 0, FEMALE: 0 });
const [ITCounts, setITCounts] = useState({ MALE: 0, FEMALE: 0 });
const [ECECounts, setECECounts] = useState({ MALE: 0, FEMALE: 0 });
const [EECounts, setEECounts] = useState({ MALE: 0, FEMALE: 0 });
const [AEIECounts, setAEIECounts] = useState({ MALE: 0, FEMALE: 0 });

function resetToDefault() {
    setSelectDisabled(false);
    setAnalyzeDisabled(true);
    setDownloadDisabled(true);
    setNumberOfRows("-");
    setNumberOfColumns("-");
    setFileSize("0 byte");
    setPlaceablePercentage("-");
    setPlaceableColor("");
    setPlaceableCount("-");
    setAnalyzedFile(null);
    setFemaleCount("-");
    setFemalePercentage("-");
    setMaleCount("-");
    setMalePercentage("-");
    setAnalyzedFile(null);
    setCSECounts({ MALE: 0, FEMALE: 0 });
    setITCounts({ MALE: 0, FEMALE: 0 });
    setECECounts({ MALE: 0, FEMALE: 0 });
    setEECounts({ MALE: 0, FEMALE: 0 });
```

```
    setAEIECounts({ MALE: 0, FEMALE: 0 });
}

function handleSignout(){
    setOpenBackdrop(true);
    axios.post('http://endpoint/v1/auth/sign_out', {
        token: Cookies.get("token")
        })
        .then(function (response) {
            if(response.status === 200){
                setSignoutState(true);
                setSnackbarMessage("Sign Out Successful!");
                setOpenSnackbar(true);
            }
        })
        .catch((error) => {
            if (error.response) {
                 if (error.response.status === 404){
                    setSignoutState(true);
                    setSnackbarMessage("Session Expired!");
                    setOpenSnackbar(true);
                 } else if (error.response.status === 500){
                    setSnackbarMessage("Server error. Please try again later.");
                    setOpenSnackbar(true);
                 }
            }else{
                setSnackbarMessage("Network error. Please check your connection.");
                setOpenSnackbar(true);
            }
            setOpenBackdrop(false);
        });
}
```

```
function handleSelectButton(event){
    const file = event.target.files[0];
    if(file){
        resetToDefault();
        setSelectFile(file);
        setFileName(file.name);
        setSelectButtonText("re-Select");
        setFileSize(file.size + " bytes");
        getRowColumnCount(file);
        setUploadDisabled(false);
    }else{
        setFileName("NO FILE SELECTED.");
    }
}

function handleUploadButton(){
    setDownloadDisabled(true);
    setSelectDisabled(true);
    setOpenBackdrop(true);
    const formData = new FormData();
    formData.append("file", selectFile);
    formData.append("token", Cookies.get("token"));
    axios.post('http://endpoint/v1/file/upload', formData, {
        headers: {
            'Content-Type': 'multipart/form-data',
        },
    })
        .then(function (response) {
            if(response.status === 200){
                setOpenBackdrop(false);
                setSnackbarMessage("Upload Successful!");
```

```
            setOpenSnackbar(true);
            setAnalyzeDisabled(false)
            setUploadDisabled(true);
        }
    })
    .catch((error) => {
        if (error.response) {
            if (error.response.status === 400 || error.response.status === 406){
                setUploadErrorMessage("Incompatible request.");
                setUploadError(true);
            } else if (error.response.status === 404){
                setSignoutState(true);
                setSnackbarMessage("Session Expired!");
                setOpenSnackbar(true);
            } else if (error.response.status === 500){
                setSnackbarMessage("Server error. Please try again later.");
                setOpenSnackbar(true);
            }
        }else{
            setSnackbarMessage("Network error. Please check your connection.");
            setOpenSnackbar(true);
        }
        setOpenBackdrop(false);
    });
}

function handleAnalyzeButton(){
    setOpenBackdrop(true);
    setAnalyzeDisabled(true);
    axios.get('http://endpoint/v1/file/get_analysis', {
        params: { token: Cookies.get('token') },
        responseType: 'blob',
```

```
})
.then((response) => {
    if(response.status === 200) {
        const blob = new Blob([response.data], {
            type: 'application/octet-stream',
        });
        getPlaceableRowCount(blob);
        getPlaceableGenderCount(blob);
        getStreamGenderCounts(blob)
        setAnalyzedFile(blob);
        setOpenBackdrop(false);
        setSnackbarMessage("Analysis complete!");
        setOpenSnackbar(true);
        setDownloadDisabled(false);
        setSelectDisabled(false);
    }
})
.catch((error) => {
    if (error.response) {
        if (error.response.status === 404 || error.response.status === 406){
            setSignoutState(true);
            setSnackbarMessage("Session Expired!");
            setOpenSnackbar(true);
        } else if (error.response.status === 500){
            setSnackbarMessage("Server error. Please try again later.");
            setOpenSnackbar(true);
        }
    }else{
        setSnackbarMessage("Network error. Please check your connection.");
        setOpenSnackbar(true);
    }
    setOpenBackdrop(false);
```

```
        });
    }

    function getRowColumnCount(file){
        const reader = new FileReader();
        reader.onload = async (e) => {
            const arrayBuffer = e.target.result;
            const workbook = XLSX.read(arrayBuffer, { type: 'array' });
            const sheet = workbook.Sheets[workbook.SheetNames[0]];
            const jsonData = XLSX.utils.sheet_to_json(sheet, { header: 1 });

            const numberOfRows = jsonData.length > 1 ? jsonData.length-1 : 0;
            const numberOfColumns = jsonData[0]?.length || 0;
            setNumberOfRows(numberOfRows.toString());
            setNumberOfColumns(numberOfColumns);
        };
        reader.readAsArrayBuffer(file);
    }

    function getPlaceableRowCount(file){
        const reader = new FileReader();
        reader.onload = async (e) => {
            const arrayBuffer = e.target.result;
            const workbook = XLSX.read(arrayBuffer, { type: 'array' });
            const sheet = workbook.Sheets[workbook.SheetNames[0]];
            const jsonData = XLSX.utils.sheet_to_json(sheet, { header: 1 });
            const headerRow = jsonData[0];
            const placeabilityIndex = headerRow.indexOf('Placeability');

            if (placeabilityIndex === -1) {
                setSnackbarMessage("Error: 'Placeability' column was not found!");
                setOpenSnackbar(true);
```

```
            return;
        }

        const placeableCount = jsonData.slice(1).reduce((count, row) => {
            return row[placeabilityIndex] === 'Placeable' ? count + 1 : count;
        }, 0);

        setPlaceableCount(placeableCount);
        setPlaceablePercentage((Math.round((placeableCount / numberOfRows) *
100)).toString());
    };
    reader.readAsArrayBuffer(file);
  }

  function getPlaceableGenderCount(file) {
    const reader = new FileReader();
    reader.onload = async (e) => {
        const arrayBuffer = e.target.result;
        const workbook = XLSX.read(arrayBuffer, { type: 'array' });
        const sheet = workbook.Sheets[workbook.SheetNames[0]];
        const jsonData = XLSX.utils.sheet_to_json(sheet, { header: 1 });
        const headerRow = jsonData[0];
        const placeabilityIndex = headerRow.indexOf('Placeability');
        const genderIndex = headerRow.indexOf('Gender');

        if (placeabilityIndex === -1) {
            setSnackbarMessage("Error: 'Placeability' column was not found!");
            setOpenSnackbar(true);
            return;
        }
        if (genderIndex === -1) {
            setSnackbarMessage("Error: 'Gender' column was not found!");
            setOpenSnackbar(true);
```

```
            return;
        }

        let femaleCount = 0;
        let maleCount = 0;

        jsonData.slice(1).forEach((row) => {
            if (row[placeabilityIndex] === 'Placeable') {
                if (row[genderIndex] === 'FEMALE') {
                    femaleCount++;
                } else if (row[genderIndex] === 'MALE') {
                    maleCount++;
                }
            }
        });
        setFemaleCount(femaleCount.toString());
        setMaleCount(maleCount.toString());
        setFemalePercentage(Math.round((femaleCount/numberOfRows) * 100).toString());
        setMalePercentage(Math.round((maleCount/numberOfRows) * 100).toString());
    };
    reader.readAsArrayBuffer(file);
}

function getPlaceableColor(){
    if(placeablePercentage === "-"){
        setPlaceableColor('#b8cdcf');
    } else if(placeablePercentage <= 33){
        setPlaceableColor('#ff2929');
    } else if(placeablePercentage <= 66){
        setPlaceableColor('#edcf26');
    } else{
        setPlaceableColor('#26ed44');
```

```
    }
}

function handleSnackbarClose(){
    if(signoutState){
        Cookies.remove("token");
        Cookies.remove("name");
    }
    setOpenSnackbar(false);
}

function handleDownloadButton(){
    if(analyzedFile){
        const link = document.createElement('a');
        link.href = URL.createObjectURL(analyzedFile);
        link.download = 'Analytics.xlsx';
        link.click();
    }
}

function getStreamGenderCounts(file) {
    const reader = new FileReader();
    reader.onload = async (e) => {
        const arrayBuffer = e.target.result;
        const workbook = XLSX.read(arrayBuffer, { type: 'array' });
        const sheet = workbook.Sheets[workbook.SheetNames[0]];
        const jsonData = XLSX.utils.sheet_to_json(sheet, { header: 1 });
        const headerRow = jsonData[0];
        const streamIndex = headerRow.indexOf('Stream');
        const genderIndex = headerRow.indexOf('Gender');
        const placeabilityIndex = headerRow.indexOf('Placeability');
```

```
        if (streamIndex === -1 || genderIndex === -1 || placeabilityIndex === -1) {
            setSnackbarMessage("Error: 'Stream', 'Gender', or 'Placeability' column was not found!");
            setOpenSnackbar(true);
            return;
        }

        const streamGenderCounts = {
            CSE: { MALE: 0, FEMALE: 0 },
            IT: { MALE: 0, FEMALE: 0 },
            ECE: { MALE: 0, FEMALE: 0 },
            EE: { MALE: 0, FEMALE: 0 },
            AEIE: { MALE: 0, FEMALE: 0 }
        };

        jsonData.slice(1).forEach(row => {
            const placeability = row[placeabilityIndex];
            if (placeability !== 'Placeable') return;

            const stream = row[streamIndex];
            const gender = row[genderIndex];

            if (streamGenderCounts[stream] && streamGenderCounts[stream][gender] !== undefined) {
                streamGenderCounts[stream][gender]++;
            }
        });
        setCSECounts(streamGenderCounts.CSE);
        setITCounts(streamGenderCounts.IT);
        setECECounts(streamGenderCounts.ECE);
        setEECounts(streamGenderCounts.EE);
        setAEIECounts(streamGenderCounts.AEIE);
    };
```

```
    reader.readAsArrayBuffer(file);
}

function handleAboutusButton(){
    navigate('/AboutUs');
}

const chartSetting = {
    yAxis: [
        {
            label: 'Placeable Students Count'
        }
    ],
    sx: {
        [`.${axisClasses.left} .${axisClasses.label}`]: {
            transform: 'translate(-20px, 0)',
        },
    },
};

useEffect(() => {
    if(Cookies.get('token') === undefined){
        navigate('/');
    }
});

useEffect(() =>{
    setUsername(Cookies.get('name'));
}, []);

useEffect(() =>{
    getPlaceableColor();
```

```
}, [placeablePercentage]);

return (
  <>
    <Box sx={{
      width: '100vw',
      height: '100vh',
      backgroundColor: '#4A628A'
      }}>
      {(<Backdrop
        sx={(theme) => ({ color: '#fff', zIndex: theme.zIndex.drawer + 1 })}
        open={openBackdrop}
      >
        <CircularProgress color="inherit" />
      </Backdrop>)}
      <Snackbar
        anchorOrigin={{ vertical: 'bottom', horizontal: 'right' }}
        open={openSnackbar}
        onClose={handleSnackbarClose}
        autoHideDuration={2000}
        message={snackbarMessage}
      />
      <Box sx={{
        display: 'flex',
        flexDirection: 'row',
        alignItems: 'center',
        width: '100vw',
        height: '14vh',
        backgroundColor: '#4A628A'
      }}>
        <Typography sx={{
          fontSize: '3em',
```

```
    marginLeft: '3vw',
    fontWeight: 100,
    color: '#DFF2EB',
}}>
    RCCIIT
</Typography>
<Typography sx={{
    fontSize: '3em',
    marginLeft: '1vw',
    color: '#DFF2EB',
}}>
    PDA
</Typography>
<Box sx={{
    width: '100vw',
    height: '14vh',
    alignItems: 'center',
    display: 'flex',
    flexDirection: 'row',
    justifyContent: 'flex-end',
    marginRight: '3vw'
}}>
    <Typography sx={{
        fontSize: '1.3em',
        marginLeft: '1vw',
        color: '#DFF2EB',
    }}>
        Hello, <span style={{fontWeight: 'bold'}}>{username}</span>
    </Typography>
    <Tooltip title="Good Bye!">
        <Button
            variant="contained"
```

```
                component="label"
                onClick={handleSignout}
                sx={{
                  marginLeft: '2vw',
                  backgroundColor: '#800000',
                  boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)',
                  '&:hover': {
                    backgroundColor: '#d40000',
                  }
                }}>
                <LogoutIcon />
              </Button>
            </Tooltip>
          </Box>
        </Box>
        <Box sx={{
          display: 'flex',
          width: '100vw',
          height: '78vh',
          backgroundColor: '#4A628A'
        }}>
          <Box sx={{ width: '30vw',
            height: '72vh',
            display: 'flex',
            flexDirection: 'column',
            margin: '5vh',
            marginTop: '0vh',
            marginRight: '2.5vh',
            borderRadius: '10px',
            justifyContent: 'flex-start',
            backgroundColor: '#7AB2D3'
          }}>
```

```
<Box sx={{
    display: 'flex',
    flexDirection: 'row',
    alignItems: 'center',
    justifyContent: 'flex-start',
    width: '90%',
    height: '10vh',
    margin: '5%',
    marginBottom: '2.5%',
    borderRadius: '10px',
    backgroundColor: '#ccd8d9',
    boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)'
}}>
    <Button
        variant="contained"
        component="label"
        disabled={selectDisabled}
        sx={{
            marginLeft: '1vw',
            backgroundColor: '#303e57',
            boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)',
            '&:hover': {
                backgroundColor: '#4A628A',
            }
        }}>
        {selectButtonText}
        <input
            type="file"
            accept=".xlsx, .xls"
            onChange={handleSelectButton}
            hidden
        />
```

```
    </Button>
    <Typography
      color="#303e57"
      sx={{
        marginLeft: '2vw',
        fontWeight: 'bold',
      }}>
      {fileName}
    </Typography>
  </Box>
  <Box sx={{
    display: 'flex',
    flexDirection: 'column',
    padding: '5%',
    width: '80%',
    flexGrow: 1,
    margin: '5%',
    marginTop: '2.5%',
    marginBottom: '2.5%',
    borderRadius: '10px',
    backgroundColor: '#ccd8d9',
    boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)'
  }}>
    <Typography
      color="#303e57"
      sx={{
        fontWeight: 'bold',
        margin: '1%'
      }}>
      FILE SIZE: {fileSize}
    </Typography>
    <Typography
```

```
                color="#303e57"
                sx={{
                    fontWeight: 'bold',
                    margin: '1%'
                }}>
                NO. OF ROWS: {numberOfRows}
            </Typography>
            <Typography
                color="#303e57"
                sx={{
                    fontWeight: 'bold',
                    margin: '1%'
                }}>
                NO. OF COLUMNS: {numberOfColumns}
            </Typography>
            <Typography
                color='#c96800'
                sx={{
                    fontWeight: 'bold',
                    margin: '1%',
                    marginTop: '30%'
                }}>
                CAUTION: Following fields are required: Gender, 10th Marks, 12th Marks, Stream and CGPA.
            </Typography>
        </Box>
        <Box sx={{
            display: 'flex',
            alignItems: 'center',
            justifyContent: 'flex-end',
            width: '100%',
            height: '10vh',
            borderRadius: '10px',
```

```
            marginBottom: '1vh',
            backgroundColor: '#7AB2D3'
        }}>
            {uploadError && (<Typography sx={{
                fontSize: '1em',
                marginLeft: '1vw',
                fontWeight: 400,
                color: '#ffffff',
                backgroundColor: '#d40000',
                width: '62%',
                padding: '1%'
            }}>
                {uploadErrorMessage}
            </Typography>)}
            <Button
                variant="contained"
                disabled={uploadDisabled}
                component="label"
                onClick={handleUploadButton}
                sx={{
                    margin: '1.5vw',
                    backgroundColor: '#303e57',
                    boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)',
                    '&:hover': {
                        backgroundColor: '#e06c00',
                    }
                }}>
                Upload
            </Button>
        </Box>
    </Box>
    <Box sx={{ width: '65vw',
```

```
        height: '72vh',
        margin: '5vh',
        marginTop: '0vh',
        marginLeft: '2.5vh',
        borderRadius: '10px',
        backgroundColor: '#7AB2D3'
    }}>
        <Box sx={{
            display: 'flex',
            flexDirection: 'row',
            justifyContent: 'space-evenly',
            weight: '100%',
            height: '24vh',
            borderRadius: '10px',
            backgroundColor: '#7AB2D3'
        }}>
            <Box sx={{
                display: 'flex',
                width: '10vw',
                marginLeft: '1.2vw',
                flexDirection: 'column',
                justifyContent: 'center',
                borderRadius: '10px',
                backgroundColor: '#7AB2D3'
            }}>
                <Button
                    variant="contained"
                    component="label"
                    disabled={analyzeDisabled}
                    onClick={handleAnalyzeButton}
                    sx={{
                        margin: '1vw',
```

```
            backgroundColor: '#303e57',
            boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)',
            '&:hover': {
              backgroundColor: '#e06c00',
            }
          }}>
          Analyze
        </Button>
        <Button
          variant="contained"
          component="label"
          disabled={downloadDisabled}
          onClick={handleDownloadButton}
          sx={{
            margin: '1vw',
            backgroundColor: '#303e57',
            boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)',
            '&:hover': {
              backgroundColor: '#4A628A',
            }
          }}>
          Download
        </Button>
      </Box>
      <Box sx={{
        display: 'flex',
        flexDirection: 'row',
        width: '15vw',
        height: '18vh',
        borderRadius: '10px',
        margin: '0.5vw',
        marginTop: '1.5vw',
```

```
        backgroundColor: placeableColor,
        boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)'
    }}>
        <Box sx={{
            display: 'flex',
            flexDirection: 'column',
            justifyContent: 'center',
            alignItems: 'center',
            width: '60%',
            height: '100%',
            borderRadius: '10px',
        }}>
            <Typography
                color='#333333'
                sx={{
                    fontSize: '3.6em',
                    fontWeight: '400',
                    margin: 'auto',
                    marginBottom: '0',
                    marginTop: '0',
                    height: '60%'
            }}>
                {placeablePercentage}%
            </Typography>
            <Typography
                color='#2b2b2b'
                sx={{
                    display: 'flex',
                    fontSize: '1em',
                    fontWeight: '600',
                    height: '20%',
                    justifyContent: 'center'
```

```
        }}>
        ~ PLACEABLE
    </Typography>
</Box>
<Box sx={{
    width: '40%',
    Height: '100%'
}}>
    <Box sx={{
        display: 'flex',
        width: '100%',
        height: '40%',
        borderTopRightRadius: '10px',
        alignItems: 'center',
        justifyContent: 'center',
        backgroundColor: '#ebebeb',
    }}>
        <Typography
            color='#2b2b2b'
            sx={{
                fontSize: '2em',
                fontWeight: '400',
            }}>
            {placeableCount}
        </Typography>
    </Box>
    <Box sx={{
        display: 'flex',
        flexDirection: 'column',
        width: '100%',
        height: '60%',
        borderBottomRightRadius: '10px',
```

```
            alignItems: 'center',
            justifyContent: 'center',
            backgroundColor: '#d4d4d4'
          }}>
            <Typography
              color='#2b2b2b'
              sx={{
                fontSize: '1em',
                fontWeight: '400',
              }}>
              OUT OF
            </Typography>
            <Typography
              color='#2b2b2b'
              sx={{
                fontSize: '2em',
                fontWeight: '400',
              }}>
              {numberOfRows}
            </Typography>
          </Box>
        </Box>
      </Box>
      <Box sx={{
        display: 'flex',
        flexDirection: 'row',
        width: '15vw',
        height: '18vh',
        borderRadius: '10px',
        margin: '0.5vw',
        marginTop: '1.5vw',
        backgroundColor: '#d79cff',
```

```
    boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)'
}}>
    <Box sx={{
        display: 'flex',
        flexDirection: 'column',
        justifyContent: 'center',
        alignItems: 'center',
        width: '60%',
        height: '100%',
        borderRadius: '10px',
    }}>
        <Typography
            color='#4a4a4a'
            sx={{
                fontSize: '3.6em',
                fontWeight: '400',
                margin: 'auto',
                marginBottom: '0',
                marginTop: '0',
                height: '60%'
            }}>
            {femalePercentage}%
        </Typography>
        <Typography
            color='#2b2b2b'
            sx={{
                display: 'flex',
                fontSize: '1em',
                fontWeight: '600',
                height: '20%',
                justifyContent: 'center'
            }}>
```

```
        ~ FEMALE <FemaleIcon/>
    </Typography>
</Box>
<Box sx={{
    width: '40%',
    Height: '100%'
}}>
    <Box sx={{
        display: 'flex',
        width: '100%',
        height: '40%',
        borderTopRightRadius: '10px',
        alignItems: 'center',
        justifyContent: 'center',
        backgroundColor: '#ebebeb',
    }}>
        <Typography
            color='#2b2b2b'
            sx={{
                fontSize: '2em',
                fontWeight: '400',
            }}>
            {femaleCount}
        </Typography>
    </Box>
    <Box sx={{
        display: 'flex',
        flexDirection: 'column',
        width: '100%',
        height: '60%',
        borderBottomRightRadius: '10px',
        alignItems: 'center',
```

```
            justifyContent: 'center',
            backgroundColor: '#d4d4d4'
          }}>
            <Typography
              color='#2b2b2b'
              sx={{
                fontSize: '1em',
                fontWeight: '400',
              }}>
              OUT OF
            </Typography>
            <Typography
              color='#2b2b2b'
              sx={{
                fontSize: '2em',
                fontWeight: '400',
              }}>
              {numberOfRows}
            </Typography>
          </Box>
        </Box>
      </Box>
      <Box sx={{
        display: 'flex',
        flexDirection: 'row',
        width: '15vw',
        height: '18vh',
        borderRadius: '10px',
        margin: '0.5vw',
        marginTop: '1.5vw',
        backgroundColor: '#ae9cff',
        boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)'
```

```
}}>
  <Box sx={{
    display: 'flex',
    flexDirection: 'column',
    justifyContent: 'center',
    alignItems: 'center',
    width: '60%',
    height: '100%',
    borderRadius: '10px',
  }}>
    <Typography
      color='#4a4a4a'
      sx={{
        fontSize: '3.6em',
        fontWeight: '400',
        margin: 'auto',
        marginBottom: '0',
        marginTop: '0',
        height: '60%'
      }}>
      {malePercentage}%
    </Typography>
    <Typography
      color='#2b2b2b'
      sx={{
        display: 'flex',
        fontSize: '1em',
        fontWeight: '600',
        height: '20%',
        justifyContent: 'center'
      }}>
      ~ MALE <MaleIcon/>
```

```
    </Typography>
</Box>
<Box sx={{
    width: '40%',
    Height: '100%'
}}>
    <Box sx={{
        display: 'flex',
        width: '100%',
        height: '40%',
        borderTopRightRadius: '10px',
        alignItems: 'center',
        justifyContent: 'center',
        backgroundColor: '#ebebeb',
    }}>
        <Typography
            color='#2b2b2b'
            sx={{
                fontSize: '2em',
                fontWeight: '400',
            }}>
            {maleCount}
        </Typography>
    </Box>
    <Box sx={{
        display: 'flex',
        flexDirection: 'column',
        width: '100%',
        height: '60%',
        borderBottomRightRadius: '10px',
        alignItems: 'center',
        justifyContent: 'center',
```

```
                backgroundColor: '#d4d4d4'
            }}>
                <Typography
                    color='#2b2b2b'
                    sx={{
                        fontSize: '1em',
                        fontWeight: '400',
                    }}>
                    OUT OF
                </Typography>
                <Typography
                    color='#2b2b2b'
                    sx={{
                        fontSize: '2em',
                        fontWeight: '400',
                    }}>
                    {numberOfRows}
                </Typography>
            </Box>
        </Box>
    </Box>
</Box>
<Box sx={{
    display: 'flex',
    flexDirection: 'row',
    borderRadius: '10px',
    height: '62%',
    width: '95.5%',
    margin: '1.5vw',
    marginTop: '0',
    backgroundColor: '#ccd8d9',
    boxShadow: '0px 4px 10px rgba(0, 0, 0, 0.25)'
```

```
                }}>
                    <Box sx={{
                        height: '95%',
                        width: '40%',
                        borderRadius: '10px',
                        backgroundColor: '#f5f5f5',
                        margin: '0.5vw',
                        display: 'flex',
                        justifyContent: 'left',
                        alignItems: 'center'
                    }}>
                        <PieChart
                            series={[
                                {
                                    data: [
                                        { id: 0, value: placeablePercentage, label: 'Placeable-%' },
                                        { id: 1, value: (100 - parseInt(placeablePercentage)), label:
'Unplaceable-%' },
                                    ],
                                    innerRadius: 15,
                                    paddingAngle: 3,
                                    cornerRadius: 6,
                                    cx: '40%',
                                    cy: '50%',
                                    highlightScope: { fade: 'global', highlight: 'item' },
                                    faded: { innerRadius: 30, additionalRadius: -30, color: 'gray' }
                                },
                            ]}
                            width={400}
                            height={200}
                        />
                    </Box>
                    <Box sx={{
```

```
                    height: '95%',
                    width: '60%',
                    borderRadius: '10px',
                    backgroundColor: '#f5f5f5',
                    margin: '0.5vw',
                    display: 'flex',
                    justifyContent: 'center',
                    alignItems: 'center',
                    paddingLeft: '0.8vw',
                }}>
                    <BarChart
                        xAxis={[{ scaleType: 'band', data: ['CSE', 'IT', 'ECE', 'EE', 'AEIE'] }]}
                        series={[
                            {data: [CSECounts.FEMALE, ITCounts.FEMALE,
ECECounts.FEMALE, EECounts.FEMALE, AEIECounts.FEMALE],
                                label: 'FEMALE', color: '#b03bff' },
                            {data: [CSECounts.MALE, ITCounts.MALE, ECECounts.MALE,
EECounts.MALE, AEIECounts.MALE],
                                label: 'MALE', color: '#471fff' }]}
                        width={500}
                        height={290}
                        borderRadius={5}
                        {...chartSetting}
                    />
                </Box>
            </Box>
        </Box>
        <Box sx={{
            display: 'flex',
            flexDirection: 'row',
            width: '100vw',
            height: '8vh',
```

```
            backgroundColor: '#7AB2D3',
            alignItems: 'center'
          }}>
            <Typography sx={{
              fontSize: '0.8em',
              fontWeight: '900',
              marginLeft: '3vw'
            }}>
              {'Copyright © RCCIIT PDA Team '}
              {new Date().getFullYear()}
              {'.'}
            </Typography>
            <Box sx={{
              display: 'flex',
              flexGrow: 1,
              justifyContent: 'flex-end'
            }}>
              <Button
                onClick={handleAboutusButton}
                sx={{
                  marginRight: '3vw',
                  color: '#ffffff',
                  backgroundColor: '#7AB2D3',
                  fontWeight: '900'
                }}
                >
                About Us
              </Button>
            </Box>
          </Box>
        </Box>
      </>
```

```
    );
  }

  export default Home;
```

6) FileService.java

```
  public ResponseEntity<?> getAnalysisService(String token) {
      Logger.info("get-analysis | attempt | token: " + token);

      if (token == null || token.isBlank()) {
        Logger.error("\tfailed | parameter error -> token");
        return new ResponseEntity<>(HttpStatus.valueOf(406));
      }

      int maxRetries = 60;
      int retryCount = 0;

      try {
        List<String> storedTokens = userRepository.getAllTokens();
        for (String storedToken : storedTokens) {
          if (token.equals(storedToken)) {

            Logger.info("Valid token provided. Checking for analysis file.");

            String analysisFileName = "Analytics." + (fileExtension != null ? fileExtension :
"xlsx");
            Path filePath = Paths.get(OUTPUT_DIR, analysisFileName);

            while (retryCount < maxRetries) {
              if (Files.exists(filePath) && Files.isReadable(filePath)) {
                Logger.info("Analysis file found: " + filePath);
                try {
```

```
                    FileSystemResource fileResource = new
FileSystemResource(filePath.toFile());
                    HttpHeaders headers = new HttpHeaders();

headers.setContentType(MediaType.APPLICATION_OCTET_STREAM);
                    headers.setContentDispositionFormData("attachment",
analysisFileName);

                    Logger.info("Returning analysis file to client: " + analysisFileName);
                    return new ResponseEntity<>(fileResource, headers, HttpStatus.OK);
                } catch (Exception e) {
                    Logger.error("Error preparing file response: ", e);
                    return new
ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
                }
            }

            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
                Logger.error("Retry interrupted: ", e);
                return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
            }

            Logger.warn("Retrying to find the analysis file: Attempt " + (retryCount + 1));
            retryCount++;
        }

        Logger.error("File not found after retries: " + analysisFileName);
        return new ResponseEntity<>(HttpStatus.NOT_FOUND);
    }
}
```

```
            Logger.error("Token not found: " + token);
            return new ResponseEntity<>(HttpStatus.NOT_FOUND);
        } catch (Exception e) {
            Logger.error("Unexpected error in getAnalysisService: ", e);
            return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
        }
    }
```

7) rcciit-pda-ml.py

```
    while True:
        # Check if there are any files in the input directory
        input_files = [f for f in os.listdir(input_dir) if os.path.isfile(os.path.join(input_dir, f))]

        if input_files:
            for file_name in input_files:
                input_file_path = os.path.join(input_dir, file_name)

                try:
                    # Determine the file extension
                    file_extension = os.path.splitext(file_name)[1]

                    # Convert Excel to DataFrame
                    test_data = pd.read_excel(input_file_path)

                    # Save original 'Sl No.' for re-adding later (exclude original 'DOB' to prevent
duplication)
                    original_columns = test_data[['Sl No.']] if 'Sl No.' in test_data.columns else None

                    # Remove 'Sl No.' from test_data to prevent duplication after re-adding
                    test_data = test_data.drop(['Sl No.'], axis=1, errors='ignore')

                    if 'DOB' in test_data.columns:
                        # Check if 'DOB' column contains numeric data (likely Excel serial dates)
```

```
                if pd.api.types.is_numeric_dtype(test_data['DOB']):
                    # Convert Excel serial date to datetime and format as 'd-m-Y'
                    test_data['DOB'] = pd.to_datetime(test_data['DOB'], unit='d', origin='1899-12-
30')
                    test_data['DOB'] = test_data['DOB'].dt.strftime('%d-%m-%Y')
                # If 'DOB' is already datetime, just reformat
                elif pd.api.types.is_datetime64_dtype(test_data['DOB']):
                    test_data['DOB'] = test_data['DOB'].dt.strftime('%d-%m-%Y')

            # Data Preprocessing for test data
            test_data['Gender'] = gender_encoder.transform(test_data['Gender'])
            test_data['Stream'] = stream_encoder.transform(test_data['Stream'])

            # Impute missing values
            imputer = SimpleImputer(strategy='most_frequent')
            test_data = pd.DataFrame(imputer.fit_transform(test_data),
columns=test_data.columns)

            # Feature Scaling
            test_data_scaled = scaler.transform(test_data.drop('DOB', axis=1, errors='ignore'))

            # Making predictions for test data
            predictions = model.predict(test_data_scaled)

            # Decode the predictions to original values and rename column to 'Placeability'
            test_data['Placeability'] = placed_encoder.inverse_transform(predictions.astype(int))
            test_data['Gender'] =
gender_encoder.inverse_transform(test_data['Gender'].astype(int))
            test_data['Stream'] =
stream_encoder.inverse_transform(test_data['Stream'].astype(int))

            # Convert values in 'Placeability' column
            test_data['Placeability'] = test_data['Placeability'].replace({
                'Placed': 'Placeable',
```

```
                'Unplaced': 'Unplaceable'
            })

            # Re-add the original 'Sl No.' if it exists
            if original_columns is not None:
                test_data = pd.concat([original_columns.reset_index(drop=True),
test_data.reset_index(drop=True)], axis=1)

            # Save output as Excel file with the same extension as input
            output_file_path = os.path.join(output_dir, f'Analytics{file_extension}')
            test_data.to_excel(output_file_path, index=False)

            print(f"File saved successfully to: {output_file_path}")

        except Exception as e:
            print(f"An error occurred while processing {file_name}: {e}")

        finally:
            # Delete the processed input file
            os.remove(input_file_path)
            print(f"Deleted input file: {input_file_path}")

    # Wait before checking the directory again
    time.sleep(5)
```