Chapter 1: Introduction

1.1 Company Profile:

EduTech Solutions is a technology firm specializing in educational analytics, founded in 2018. With a team of 50 data scientists and engineers, the company develops AI-driven tools to enhance academic and career outcomes for educational institutions. EduTech serves over 100 colleges, focusing on predictive analytics and decision support systems.

1.2 Existing System and Need for System:

Currently, colleges rely on manual assessments or basic statistical methods to predict student placement outcomes, which are time-consuming and often inaccurate. These methods fail to account for complex relationships between academic performance (e.g., CGPA) and cognitive abilities (e.g., IQ). A machine learning-based system is needed to accurately predict placement likelihood, enabling colleges to provide targeted career guidance and improve placement rates.

1.3 Scope of Work:

The proposed Machine Learning Model will:

- Predict student placement (Placed/Not Placed) based on CGPA and IQ level.
- Provide a web-based interface for inputting student data and viewing predictions.

- Generate reports on placement trends for institutional use.
- Exclude factors like extracurricular activities or interview skills.

1.4 Operating Environment - Hardware and Software:

- Hardware: Standard PCs with Intel i5/i7 processors, 16GB RAM, 1TB storage; optional GPU (e.g., NVIDIA GTX 1660) for model training.
- Software:
  - OS: Windows 11 or Linux (Ubuntu 22.04).
  - Programming: Python 3.9, Jupyter Notebook for development.
  - Libraries: scikit-learn (machine learning), pandas (data processing), Flask (web interface), Matplotlib/Seaborn (visualization).
  - Database: SQLite for storing student data and predictions.

Chapter 2: PROPOSED SYSTEM

2.1 Proposed System:

The Student Placement Prediction System uses a supervised machine learning model (e.g., Logistic Regression or Random Forest) to predict whether a student will be placed based on their CGPA and IQ level. The system includes a web interface for inputting student details and displaying predictions, along with a reporting module for placement analytics.

2.2 Objectives of System:

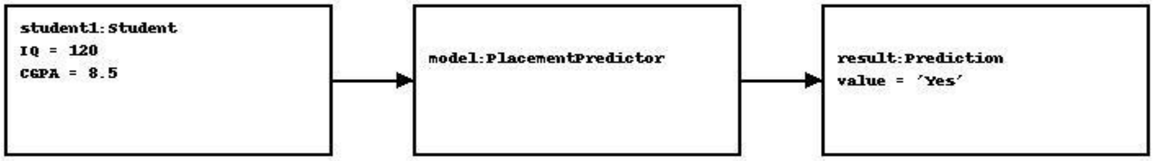
- Achieve at least 85% accuracy in predicting student placement.
- Provide a user-friendly interface for colleges to input student data.
- Generate placement trend reports to guide institutional strategies.
- Ensure scalability to handle data for up to 10,000 students.

2.3 User Requirements

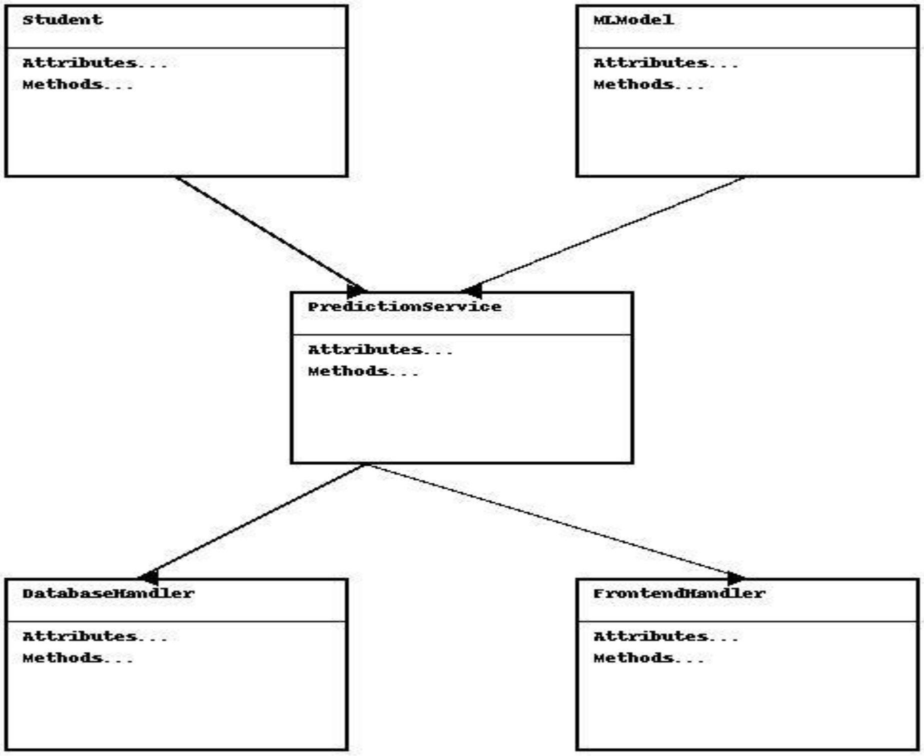
- Functional:
    - Input student CGPA and IQ to predict placement.
    - View individual and batch predictions.
    - Generate placement trend reports.
- Non-Functional:
    - System response time under 3 seconds for predictions.
    - Secure data handling with user authentication.
    - Model training time under 10 minutes for 1,000 records.

Chapter 3: ANALYSIS & DESIGN

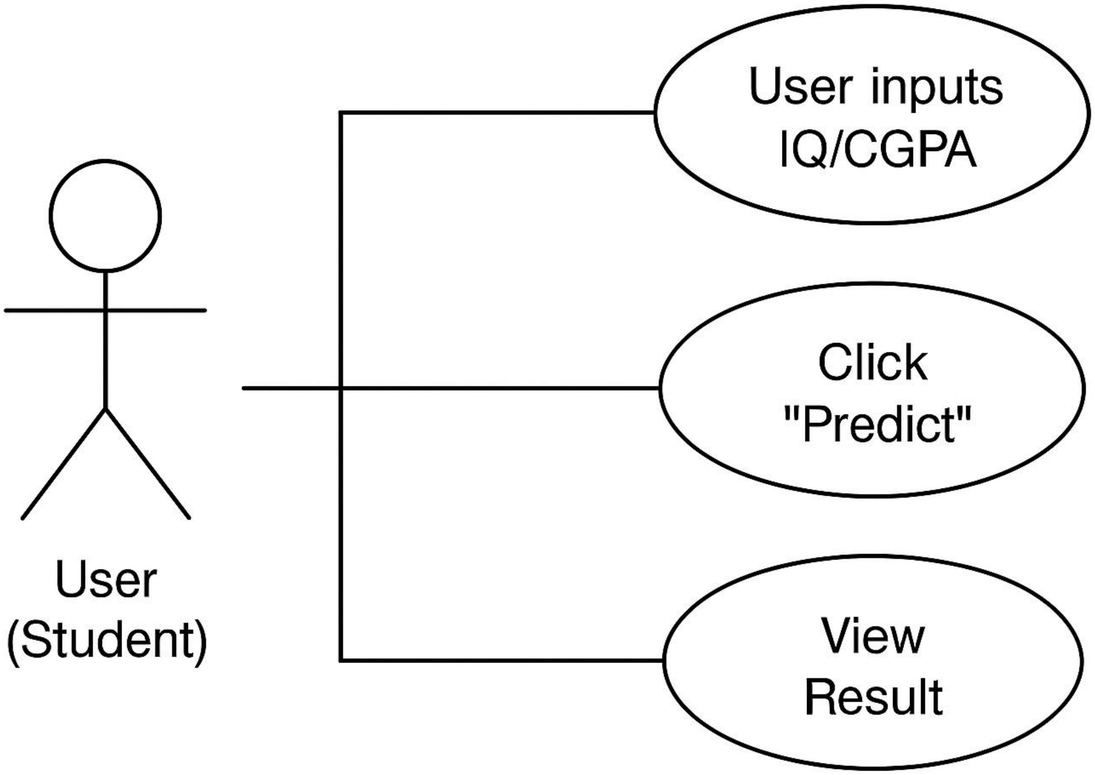
3.1 Object Diagram



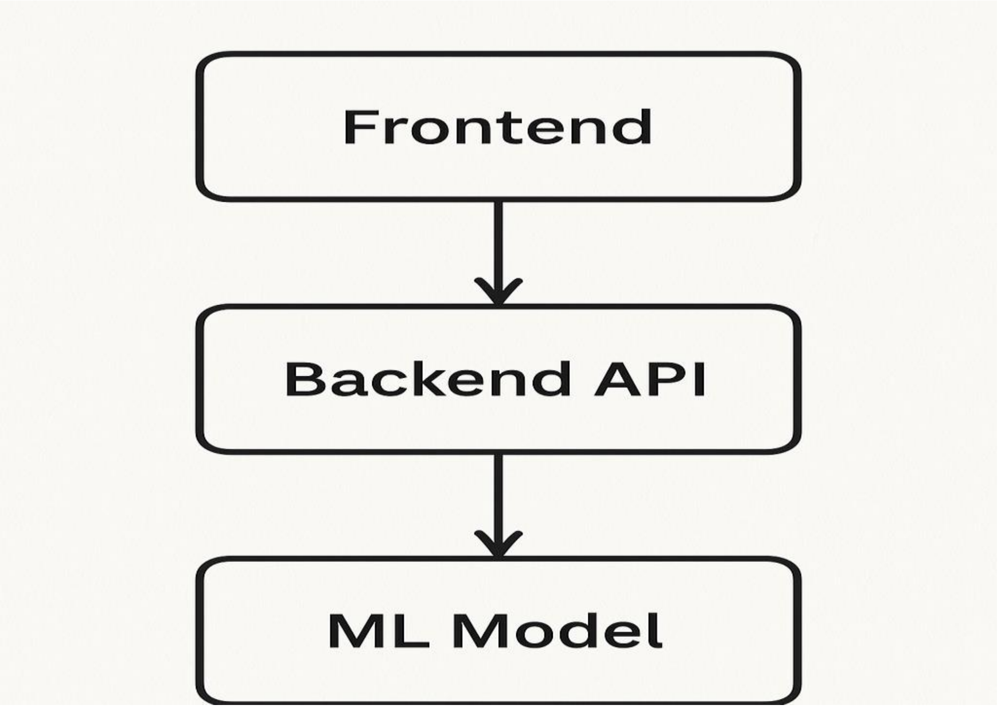
3.2 Class Diagram :



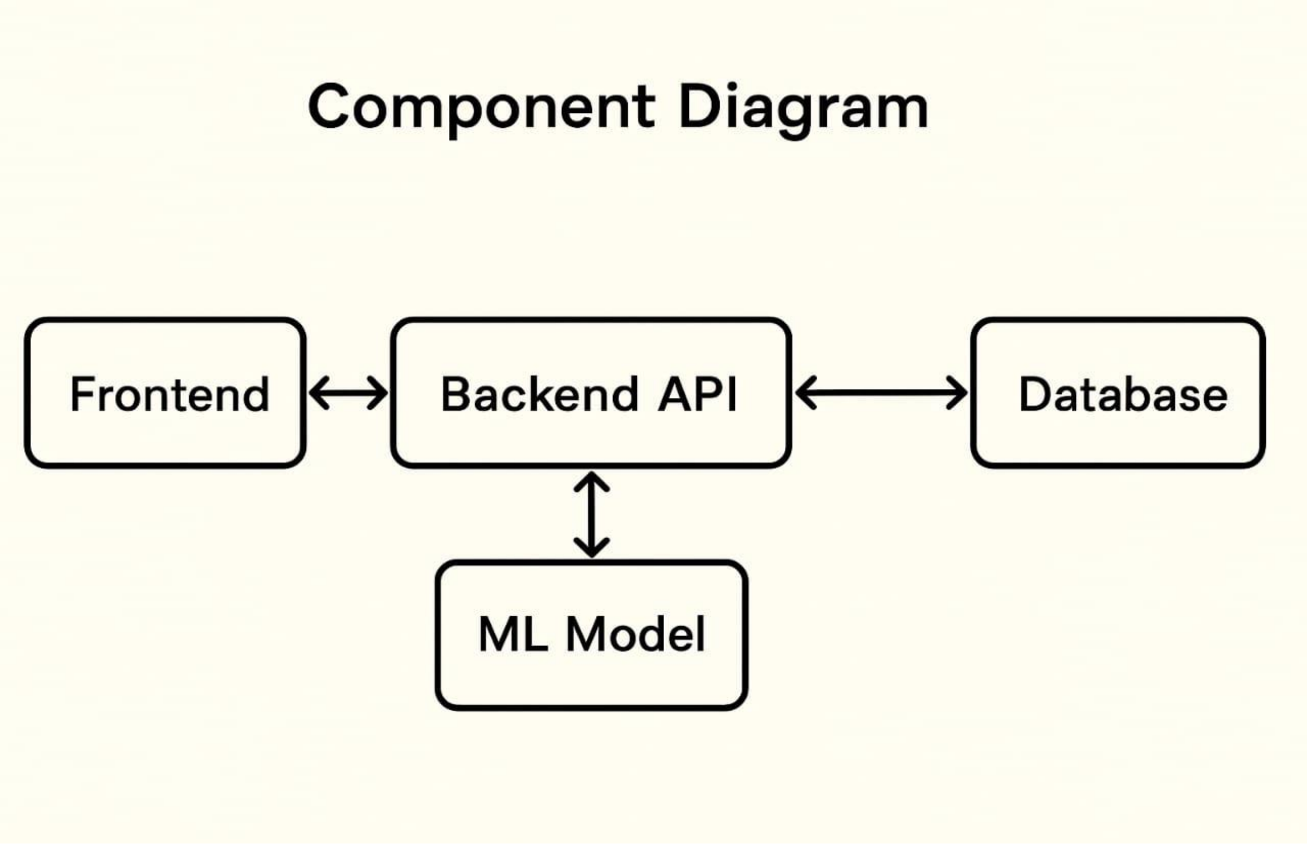
3.3 Use Case Diagram:



3.4 Module Hierarchy Diagram:



3.5 Component Diagram:



3.6 Module Specification:

1. User Interface Module (Frontend)
Description: Interface for users to input IQ and CGPA and view results.

Technologies: HTML, CSS, JavaScript (or React/Angular).

Functions:

Capture user input.

Send data to backend via HTTP requests.

Display prediction results returned from the backend.

2. API/Backend Module
Description: Acts as a bridge between the frontend and the ML model.

Technologies: Flask/Django (Python) or Node.js.

Functions:

Receive IQ and CGPA from the frontend.

Validate input data.

Pass data to the machine learning model.

Return the prediction result to the frontend.

3. Machine Learning Module
Description: Contains the prediction logic using a trained ML model.

Technologies: Python, Scikit-learn/TensorFlow/Keras.

Functions:

Load the pre-trained model.

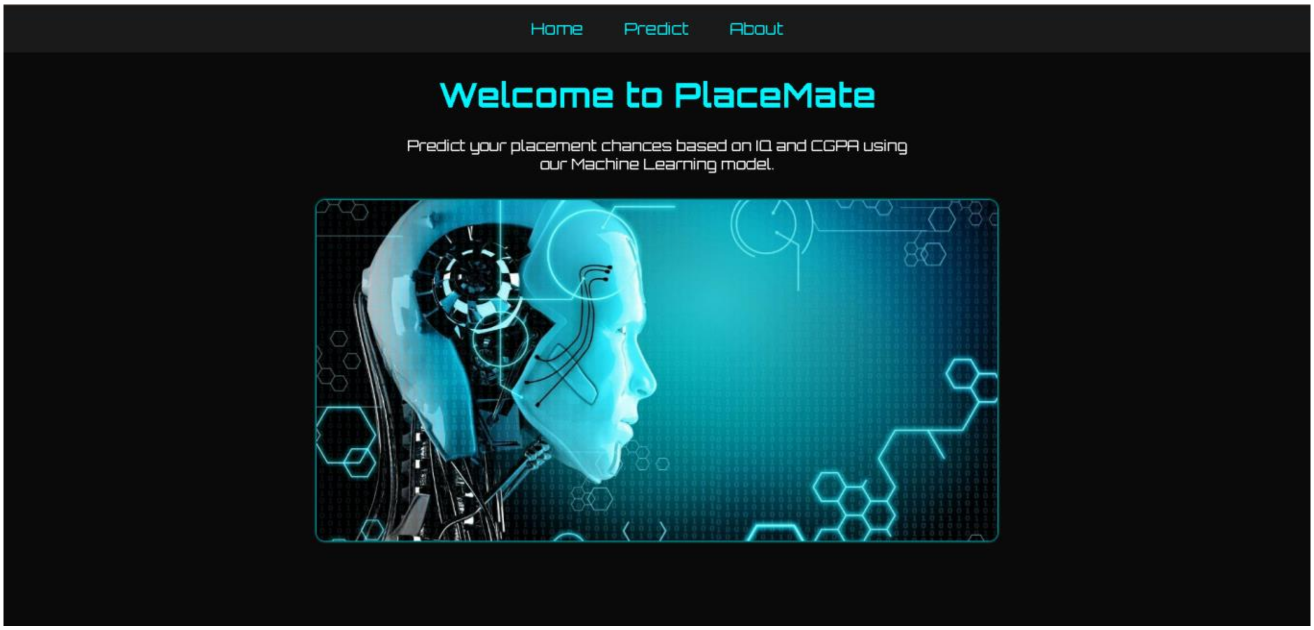
Process and normalize input features.

Make predictions (Placed/Not Placed).

Return prediction to the backend.

3.7 User Interface Design:

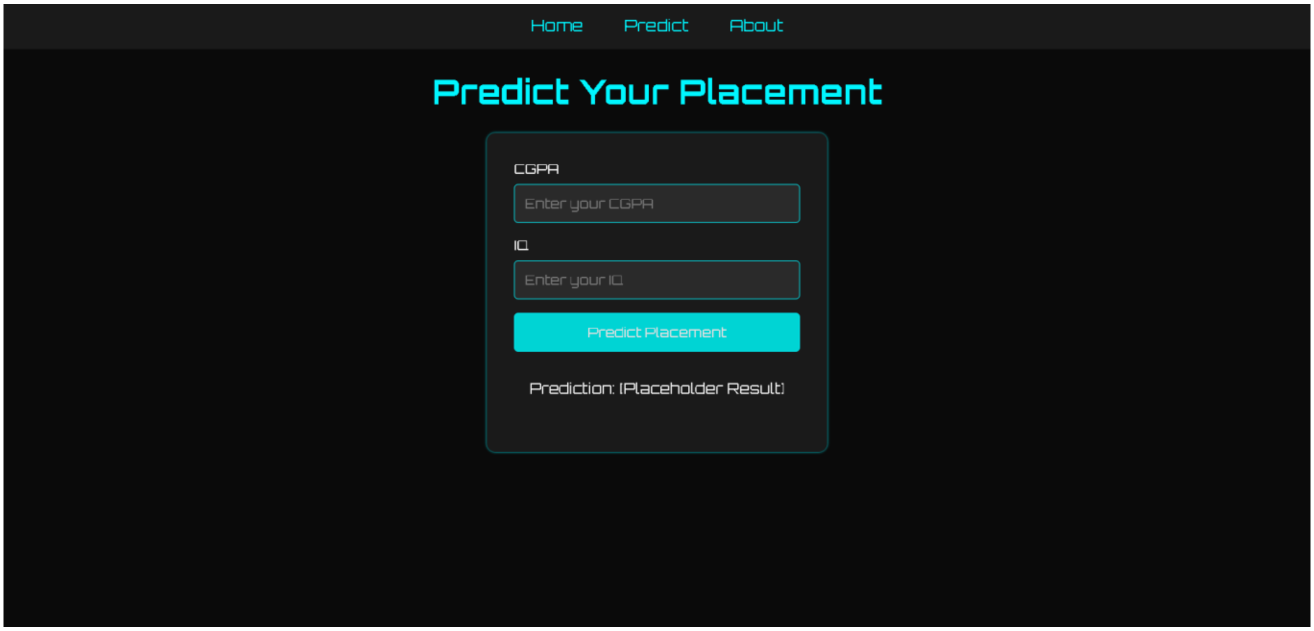
Home Page:



Welcome to PlaceMate

Predict your placement chances based on IQ and CGPA using our Machine Learning model.

Predict Page:



Predict Your Placement

CGPA

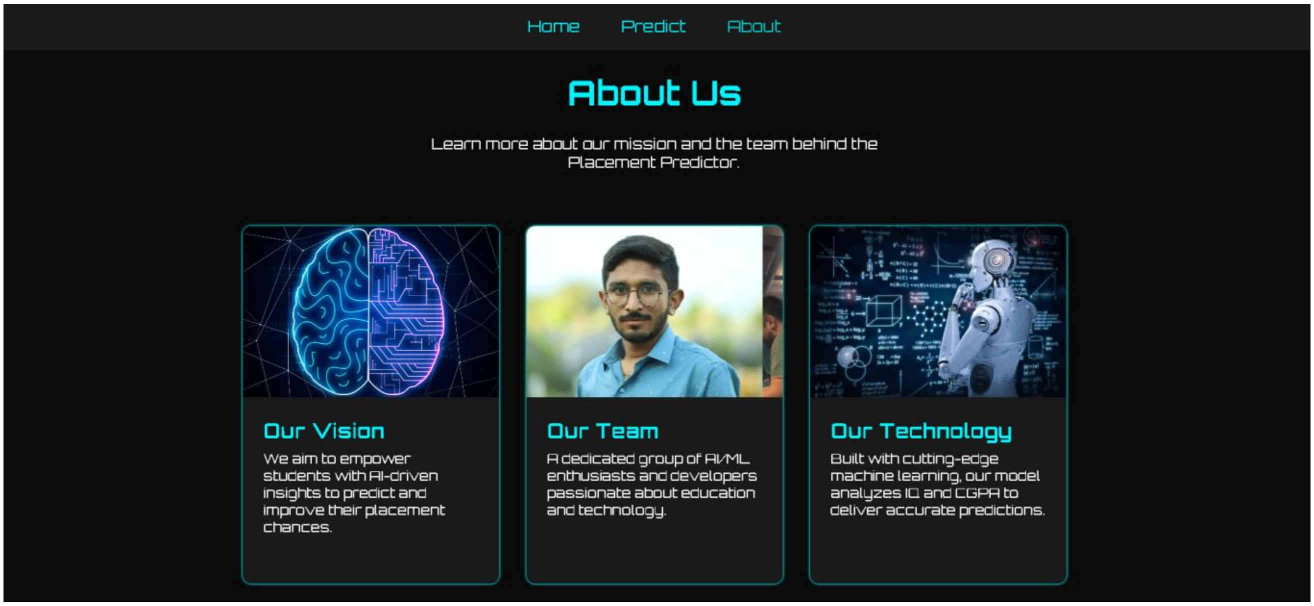
Enter your CGPA

IQ

Enter your IQ

Prediction: [Placeholder Result]

About Page:



About Us

Learn more about our mission and the team behind the Placement Predictor.

Our Vision

We aim to empower students with AI-driven insights to predict and improve their placement chances.

Our Team

A dedicated group of AI/ML enthusiasts and developers passionate about education and technology.

Our Technology

Built with cutting-edge machine learning, our model analyzes IQ and CGPA to deliver accurate predictions.

3.8 Test Procedures and Implementation:

No.	Test Condition	Description	Expected Result	Actual Output	Status
1	Home Page Load	Load the Home page	Home page should display Welcome text, slider images, and navigation bar	Home page displayed correctly with 'Welcome to PlaceMate', images, and nav bar	Pass
2	Predict Placement - Empty Inputs	Click Predict without entering CGPA and IQ	System should show validation error or no prediction should happen	No prediction, error shown in console	Pass
3	Predict Placement - Valid Inputs	Enter valid CGPA and IQ, then click Predict	System should send data to backend and show prediction result	Displayed result correctly based on prediction	Pass
4	About Page Load	Click About section from nav	About Us page with cards should load showing Vision, Team, and Technology	Page loaded correctly with all cards visible	Pass
5	Navigation Bar Links	Click Home, Predict, About links	Images should slide/animate automatically	Navigation smooth and sections visible	Pass

Chapter 4: USER MANUAL

4.1 User Manual:

Overview:

This user manual is intended to guide college staff on how to efficiently use the Placement Prediction System.

The system helps predict the placement chances of students based on their CGPA and IQ values, either individually or in batch mode.

It also allows users to generate detailed Placement Trend Reports for analysis and decision-making.

Steps to Use the System:

1. Open the Website:

- Launch a web browser such as Google Chrome, Microsoft Edge, or Mozilla Firefox.
- Enter the website URL (e.g., http://localhost:5000 if hosted locally, or your provided server address).
- Press Enter to load the site.

2. Home Page Redirection:

- Upon opening the URL, you will be automatically redirected to the Home Page.
- The Home Page will display a welcome message:

"Welcome to PlaceMate"

- It features a navigation bar at the top with options like:
    - Home
    - Predict
    - About

- It also displays a slideshow with images related to placement or education.

3. Use Prediction Menu for Individual or Batch Predictions:

- Click on the Predict menu option in the navigation bar.

Individual Prediction:

- You will see a simple form asking for two inputs:
    - CGPA: Enter the student's Cumulative Grade Point Average.
    - IQ: Enter the student's IQ score.
- After filling the fields, click the Predict Placement button.
- The system will process the data and display the prediction:
    - Student Placement: Yes (likely to get placed)
    - Student Placement: **No** (less likely to get placed)

Batch Prediction *(if implemented)*:

- (Future Enhancement) For batch uploads, you may be provided an option to upload a CSV file containing multiple student records.
- After uploading, results will be shown either in a table format or downloadable as a report.

4.2 Operations Manual / Menu Explanation:

The Placement Prediction System is designed to be simple and user-friendly.

This section explains the main operations available through the system menus and how to use each one properly.

Data Input:

Purpose: Add student data individually or upload data for a batch of students for prediction processing.

a) Individual Student Data Input:

- Navigate to the Predict page through the top navigation menu.
- Enter the CGPA (Cumulative Grade Point Average) and IQ (Intelligence Quotient) values for a single student.
- After filling the input fields, click on the Predict Placement button.
- The entered data will be sent to the backend server for real-time prediction.

Prediction:

Purpose: Predict whether students will likely be placed or not based on input data.

a) Individual Prediction:

- After submitting CGPA and IQ in the input form, the system immediately returns the placement result for that specific student.
- The result will appear as:
    - Student Placement: Yes — indicating a high probability of getting placed.
    - Student Placement: No — indicating a low probability of getting placed.

b) Batch Prediction:

- For batch uploads, once the CSV file is processed, the system will generate a placement prediction for each student in the file.
- The predictions can be viewed on-screen in a list/table format.
- Optionally, the batch results can be downloaded for record-keeping.

Reports:

Purpose: Generate insights and trends about placement outcomes based on previous predictions.

a) Placement Trend Reports:

- Navigate to the Reports section from the menu (available if implemented).

- You will have options to filter data by:
  - Department (e.g., CSE, IT, ECE)
  - Academic Year or Batch
  - CGPA Range or IQ Range
- Select your desired filters and click on Generate Report.

b) Report Output:

- The system will display placement trends such as:
  - Percentage of students likely to be placed.
  - Relationship graphs between CGPA/IQ and placement chances.
  - Trends over different batches or departments.

c) Report Download:

- After generating the report, you can download it in the following formats:
  - PDF for printing or presentations.
  - Excel (.xlsx) for further data analysis.

4.3 Forms and Report Specifications:

This section defines the structure, fields, and purpose of all forms and reports in the **Placement** Prediction System.

Following these specifications ensures consistent data input and accurate reporting.

Forms

a) Add Form (Individual Entry)

Purpose**:** To collect placement prediction inputs for a single student.

Fields:

Field Name	**Type**	**Description**	**Validation**
CGPA	Number (Decimal)	Student's Cumulative Grade Point Average (e.g., 8.5)	Must be between 0.0 and 10.0
IQ	Number (Decimal)	Student's Intelligence Quotient score (e.g., 120)	Must be a positive value

Behaviour**:**

- Both fields are **required** to submit the form.
- When the user clicks **Predict Placement**, the system immediately processes the input and displays the prediction result.
- Errors are shown if input fields are empty or invalid.

Reports:

a) Placement Trend Report:

Purpose: Visualizes overall placement success trends based on CGPA and IQ ranges.

Contents:

- Graphs and Charts like:
  - CGPA Range vs Placement Percentage (e.g., 6.0-7.0, 7.1-8.0, etc.)
  - IQ Range vs Placement Percentage
  - Overall Placement Success Rate

- Key Insights such as:
  - "Students with CGPA above 8.0 have a 90% placement chance."
  - "IQ above 120 correlates with 85% placement success."

Format Options:

- View online (HTML page).
- Export as PDF or Excel.

Drawbacks and Limitations:

While the Placement Prediction System offers a simple and efficient way to estimate placement chances, it is important to recognize its current limitations.

Understanding these drawbacks helps users interpret the predictions carefully and plan for future improvements.

1. Reliance Only on CGPA and IQ

Limitation**:**

The system predicts student placement outcomes based solely on two factors:

- CGPA (Cumulative Grade Point Average)
- IQ (Intelligence Quotient)

Explanation**:**

- Real-world placement outcomes depend on many more factors beyond academic performance and IQ.
- Important attributes like:
  - Technical skills (programming, software tools)
  - Soft skills (communication, teamwork, leadership)
  - Internship experience
  - Extracurricular activities
  - Certifications (AWS, Azure, Python, etc.)
  - Interview performance and group discussions

    are not considered in this model.

Impact**:**

- Predictions might not fully capture a student's true employability.
- High CGPA and IQ students may still struggle if they lack real-world skills.
- Some students with lower CGPA but strong certifications or communication skills may succeed but be wrongly predicted as unlikely to be placed.

2. No Real-Time Data Integration with College Databases

Limitation**:**

The system currently does not connect to any live databases or student management systems used by colleges.

Explanation**:**

- Student data (CGPA, IQ, personal details) must be entered manually or uploaded manually via a CSV file.
- There is no automated synchronization with:
    - Student Information Systems (SIS)
    - Examination result servers
    - Certification achievement systems
    - Placement tracking tools (like T&P Cell platforms)

Impact**:**

- Manual entry increases the chances of human errors (wrong CGPA, wrong IQ values).
- Placement predictions may become outdated if student academic records change but are not updated in the system.
- Time-consuming for large colleges with thousands of students to constantly upload updated data.
- Batch uploads are helpful, but real-time syncing would make the system far more powerful, reliable, and efficient.

Proposed Enhancements

To improve the accuracy, scalability, and accessibility of the Placement Prediction System, the following enhancements are proposed.

These upgrades will help the system evolve from a basic predictor to a robust real-world application used by colleges and students alike.

1. Incorporate Additional Features like Resume Quality and Technical Skills

Proposal:

Expand the prediction model to include more real-world factors beyond CGPA and IQ.

Additional Features to Include:

- Resume Quality:

  Evaluate the structure, content, and presentation of a student's resume using AI-based resume analyzers or simple scoring criteria (e.g., projects included, certifications, internships listed).
- Technical Skills:

  Collect student certifications and skillsets (e.g., Python, Java, AWS Certification) and use them as additional inputs.
- Soft Skills:

  Use student self-assessments or interview evaluations to include communication, leadership, and teamwork skills.

Benefits:

- Predictions will be more realistic and holistic, considering overall employability.
- Students who work on improving their skills can see immediate improvements in their predicted chances.

Implementation Approach:

- Expand the form to collect technical skills, certifications, and resume scores.
- Retrain the Machine Learning model with the new features.

2. Upgrade to a Scalable Database (e.g., PostgreSQL) for Larger Datasets

Proposal:

Migrate student data storage from basic file handling (CSV) to a relational, scalable database system like PostgreSQL.

Why PostgreSQL?

- Handles large volumes of student records efficiently.
- Supports complex queries (e.g., retrieve students by department, CGPA range).
- Enables secure multi-user access with permissions.
- Integrates easily with web applications and APIs.

Benefits:

- Faster data retrieval and batch processing for thousands of students.
- Better data management — no more CSV upload hassles.
- Automatic updates when a student's CGPA, IQ, or placement status changes.
- Can support real-time dashboards and analytics reports.

Implementation Approach:

- Set up a PostgreSQL database with tables for Student Data, Prediction Results, and Reports.
- Connect the website backend (Flask or Django) to PostgreSQL using ORM libraries like SQLAlchemy.

3. Develop a Mobile App for On-the-Go Access

Proposal:

Create a lightweight mobile application (Android/iOS) to allow students and staff to access the Placement Prediction System anywhere, anytime.

App Features:

- Individual Prediction: Students can enter their CGPA and IQ to get real-time prediction results on their phone.
- Batch Upload (Admin only): Placement staff can upload or view batch predictions.
- Reports Access: Staff can generate and download placement trend reports directly from the app.
- Push Notifications: Notify students about prediction updates, placement drives, or report availability.

Benefits:

- Convenience: Students can quickly check their placement status without needing a laptop or PC.
- Accessibility: Placement staff can manage reports and predictions during college events or while traveling.
- Modern User Experience: Boosts the system's acceptance and usage.

Implementation Approach:

- Build a mobile app using frameworks like Flutter (cross-platform) or React Native.
- Integrate the app with the backend APIs already used by the web system.

Conclusion:

The Machine Learning Model for Student Placement Prediction is a significant step forward in assisting colleges with forecasting student placement outcomes.

By leveraging academic performance (CGPA) and intelligence measures (IQ), the system offers an accurate, fast, and user-friendly tool for colleges to better understand the employability of their students.

Key Achievements:

- Accuracy and Efficiency:

  The system quickly processes student data and delivers predictions without the need for manual analysis.

  It provides a reliable first-level estimation of a student's placement potential.

- Career Guidance Enhancement:

  Students and placement officers can use the system’s feedback to:

  - Identify students at risk of not being placed.
  - Offer targeted career counseling or training programs.
  - Help students understand the importance of academic performance and IQ development in securing job placements.

- Institutional Planning Support:

  Colleges can:

  - Use the generated Placement Trend Reports to analyze trends across different batches, departments, or academic years.
  - Make data-driven decisions for improving their placement strategies and training programs.
  - Identify the effectiveness of skill development initiatives over time.

Opportunities for Future Improvements:

While the current model is effective, there is clear potential for further enhancements to make it even more powerful and realistic.

Possible areas for improvement include:

- Incorporating Additional Features: Adding parameters such as technical certifications, internship experiences, communication skills, and resume quality would significantly improve prediction accuracy.
- Integration with College Databases: Real-time syncing with student management systems would automate data input and maintain up-to-date records.
- Mobile App Development: Extending access through mobile platforms would make predictions and reports even more accessible for students and staff.
- Advanced Analytics and AI Insights: Over time, the system could evolve to not just predict placement but recommend actions to improve a student's chances based on their profile.

Final Thought:

The Placement Prediction System represents a strong foundation for a future-ready, data-driven approach to student placement management.

By continuously refining the model and expanding its scope, colleges can empower students, optimize placement operations, and improve institutional outcomes in a competitive employment landscape.

Bibliography:

1. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning* (2nd ed.). Springer. https://web.stanford.edu/~hastie/ElemStatLearn/

2. scikit-learn Documentation (2023). Retrieved from https://scikit-learn.org/stable/documentation.html

3. Flask Documentation (2023). Retrieved from https://flask.palletsprojects.com/

4. Python Official Documentation (2023). Retrieved from https://docs.python.org/3/

5. Kaggle Documentation (2023). Retrieved from https://www.kaggle.com/docs

This is the official Kaggle documentation, which provides a wealth of resources on data science and machine learning competitions, datasets, and notebooks that can enhance practical learning.

ANNEXURE 1: Sample Program Code.

- student.html

```
<html>

<head>
  <title>Placement Predictor</title>
  <link href="https://fonts.googleapis.com/css2?family=Orbitron:wght@400;700&display=swap"
rel="stylesheet">
  <link href="style.css" rel="stylesheet">
</head>
<body>

  <nav>
    <ul>
      <li><a href="#home" class="neon-text">Home</a></li>
      <li><a href="#predict" class="neon-text">Predict</a></li>
      <li><a href="#about" class="neon-text">About</a></li>
    </ul>
  </nav>


  <div id="home" class="page">
    <h1 class="neon-text">Welcome to PlaceMate</h1>
    <p>Predict your placement chances based on IQ and CGPA using our Machine Learning
model.</p>
    <div class="slider neon-border">
      <div class="slides">
        <div class="slide">
          <img src="5.jpg">
        </div>
        <div class="slide">
          <img src="5.jpg">
        </div>
      </div>
    </div>
  </div>


  <div id="predict" class="page">
    <h2 class="neon-text">Predict Your Placement</h2>
    <div class="form-container">
      <label for="cgpa">CGPA</label>
      <input type="number" id="cgpa" name="cgpa" placeholder="Enter your CGPA" required>
      <label for="iq">IQ</label>
      <input type="number" id="iq" name="iq" step="0.1" placeholder="Enter your IQ" required>
      <input type="submit" value="Predict Placement" onclick="sendData()" id="button">
      <p style="color: #f4f4f4; margin-top: 1rem;">Prediction: [Placeholder Result]</p>
      <div id="result"></div>
    </div>
  </div>


  <div id="about" class="page">
    <h2 class="neon-text">About Us</h2>
    <p>Learn more about our mission and the team behind the Placement Predictor.</p>
    <div class="cards-container">
      <div class="card neon-border">
        <img src="Our_vision.jpg">
        <div class="card-content">
          <h3>Our Vision</h3>
          <p>We aim to empower students with AI-driven insights to predict and improve their
placement chances.</p>
        </div>
      </div>
      <div class="card neon-border">
        <div class="card-slider">
          <div class="card-slides">
            <div class="card-slide">
              <img src="maan.jpg">            </div>
            <div class="card-slide">
              <img src="om.jpg">            </div>
            <div class="card-slide">
              <img src="mustafa.jpg">            </div>
          </div>
        </div>
        <div class="card-content">
          <h3>Our Team</h3>
          <p>A dedicated group of AI/ML enthusiasts and developers passionate about education and
technology.</p>
        </div>
      </div>
      <div class="card neon-border">
        <img src="new.jpg">
        <div class="card-content">
          <h3>Our Technology</h3>
          <p>Built with cutting-edge machine learning, our model analyzes IQ and CGPA to deliver
accurate predictions.</p>
        </div>
      </div>
    </div>
  </div>
  <script>
```

```
function sendData() {
        const cgpa = parseFloat(document.getElementById('cgpa').value);
        const iq = parseFloat(document.getElementById('iq').value);

        fetch('http://localhost:5000/predict', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({ cgpa: cgpa, iq: iq })
        })
        .then(response => response.json())
        .then(data => {
            const placement = data.prediction === "1" ? "Yes" : "No";
            document.getElementById('result').innerText = "Student Placement: " + placement;
        })
        .catch(error => {
            console.error('Error:', error);
            document.getElementById('result').innerText = "Error connecting to the backend!";
        });
    }
</script>
</body>
</html>
```

- App.py

```
from flask import Flask, request, jsonify
import numpy as np
import pickle
import joblib
from flask_cors import CORS

app = Flask(__name__)
CORS(app)

# Load model and scaler
model = pickle.load(open("reg_model.pkl", "rb"))  # logistic regression model
scaler = joblib.load("reg.pkl")  # scaler used during training

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json()
    cgpa = float(data['cgpa'])
    iq = float(data['iq'])

    # Prepare data for prediction
    features = np.array([[cgpa, iq]])
    scaled_features = scaler.transform(features)

    # Predict using the model
    prediction = model.predict(scaled_features)[0]  # 0 or 1

    return jsonify({'prediction': str(prediction)})

if __name__ == '__main__':
    print("starting server")
    app.run(debug=True)
```