Student Performance Prediction System

A PROJECT REPORT

Submitted to

Dr Javed Alam.

Submitted by

ANSH CHOPRA (24MCI10105)

in partial fulfillment for the award of the degree of

MASTER OF COMPUTER APPLICATION

IN

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Chandigarh University

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to **Dr Javed Alam** , my project supervisor, for his invaluable guidance, support, and encouragement throughout the duration of this project. His expertise and insights were instrumental in the successful completion of the project, *"Student Performance Prediction System."*

I am also thankful to **Chandigarh University** for providing the resources and a conducive environment that enabled me to explore and apply algorithmic principles in solving real-world problems. This project has been a significant learning experience, enhancing both my theoretical knowledge and practical skills.

TABLE OF CONTENTS

ABSTRACT

This project aims to create a system that uses machine learning techniques to predict the final exam scores of students. By analyzing academic data like attendance rates, assignment grades, quiz scores, and midterm results, the system forecasts the final performance using a linear regression model. This tool is designed to support both students and educators by providing early indications of academic performance, offering tailored study suggestions based on predicted outcomes, and generating downloadable pdf report cards. The online platform provides a user-friendly interface that facilitates seamless interaction and visualization of the outcomes, thereby improving the overall user experience.

\

ABBREVIATIONS AND SYMBOLS

ML – Machine Learning

MAE – Mean Absolute Error

R^2 – Coefficient of Determination

CSV – Comma-Separated Values

PDF – Portable Document Format

UID – Unique Identifier

GUI – Graphical User Interface

HTML/CSS – Frontend Technologies

Flask – Python Web Framework

Linear Regression – A type of statistical model that predicts a dependent variable based on one or more independent variables.

CHAPTER 1.
INTRODUCTION

1.1 Identification of Client & Need

Educational institutions are increasingly adopting data analytics to monitor and enhance student outcomes. Both students and educators benefit from timely insights into academic performance. However, such insights are often only available after final examinations. This system addresses the growing demand for early performance prediction by analyzing continuous assessment metrics. It enables students to modify their learning approaches proactively, and assists instructors in offering timely support to those who may be at academic risk.

1.2 Relevant Contemporary Issues

Data-Driven Learning: The integration of predictive analytics into educational systems is reshaping how performance is assessed and improved.

Scalability: Manually analyzing large-scale academic data is impractical, creating a need for automated systems.

Customized Learning Paths: Leveraging predictions to tailor feedback supports more effective, individualized study plans.

1.3 Problem Identification

Students often remain unaware of their academic standing until final results are published, missing out on opportunities to improve. Simultaneously, teachers struggle to continuously monitor every student's progress, delaying intervention. Existing systems either lack accessibility or fail to offer personalized feedback along with performance prediction.

1.4 Task Identification

- Develop a model that predicts final exam scores using interim assessments.

- Provide tailored study advice based on predicted outcomes.
- Build an intuitive web interface for data input and result presentation.
- Offer downloadable report cards summarizing predictions and feedback.

1.5 Organization of the Report

The document is structured to present the background and need for the system, review existing work, describe the design process, evaluate outcomes, and propose potential enhancements.

CHAPTER 2 :
LITERATURE SURVEY

2.1 Introduction to educational performance prediction models.

The significance of predictive analytics in education has grown significantly in recent years, fueled by the expansion of data and the desire to enhance academic achievements. Predictive models aid in forecasting student performance by considering factors like attendance, assignments, quizzes, and mid-term exams. These models not only forecast student grades but also offer valuable insights into which students may require additional assistance, thereby enhancing personalized learning and improving educational strategies.

2.2 Traditional predictive models in education.

Linear regression is one of the most commonly employed predictive models in the field of education. It is highly favored for its simplicity, interpretability, and ease of implementation. Linear regression seeks to forecast a continuous outcome (e.G., final exam score) by considering one or more input features (e.G., attendance, assignments, quizzes). This approach has proven to be successful in educational performance prediction tasks, where the connections between inputs and outcomes are either linear or nearly linear.

For instance, a study conducted by tsetsi & denning (2018) showcased the efficacy of linear regression in forecasting academic performance by considering attendance and assignment scores. The findings indicated that linear regression models were capable of estimating final exam scores reasonably well, particularly when the dataset was small and uncomplicated.

2.3 Applying Machine Learning Algorithms to Predictive Analytics

While linear regression models are commonly employed, researchers have also delved into more advanced machine learning techniques, including support vector machines (svm), random forests, and neural networks, in educational settings. These models are better suited for capturing non-linear relationships in large datasets and generally

perform better in terms of accuracy when handling complex datasets with numerous features.

Zhou et al. (2020) have utilized support vector machines (svm) to predict student success in online courses. Their investigation revealed that svm outperformed conventional regression models, particularly in datasets with a large number of features. Nevertheless, the main obstacle with svm and similar algorithms is the requirement for substantial data and substantial computational resources, rendering them less feasible for smaller educational datasets.

Random forests, which create numerous decision trees and combine their predictions, are a widely used approach in education analytics. They are particularly beneficial for managing datasets that have numerous features and can automatically identify intricate relationships between variables. Lee et al. (2021) employed random forests to forecast academic performance by considering demographic information, study habits, and involvement in extracurricular activities. The results of their research showed that random forests outperformed traditional models and were successful in predicting student outcomes for extensive datasets.

Nevertheless, these intricate models often come with the drawback of decreased interpretability, making it challenging for educators and administrators to comprehend the reasoning behind the predictions. Linear regression has an advantage over other statistical methods, as its results are more easily understood and explained to non-technical stakeholders.

2.4 Implementation of corrective actions.

A crucial feature of contemporary predictive systems is their capability to offer customized feedback. Huang & Lee (2019) examined how providing personalized feedback can enhance student engagement and academic achievement. The study emphasized that when predictive models are integrated with personalized recommendations (such as study tips or learning resources), student engagement and academic performance experience substantial enhancements. This method transforms predictive systems from basic tools for estimating performance into proactive systems that actively support student learning.

Additionally, reddy & gupta (2021) demonstrated that real-time feedback based on predictions could be provided to students, enabling them to modify their study habits

at the beginning of the semester. This early intervention is crucial in preventing students from falling behind and ensures that they have the best possible learning experience.

2.5 Comprehensive methodologies and supplementary information.

While traditional academic indicators like assignments, quizzes, and mid-term scores are crucial for predicting student performance, recent research has highlighted the significance of considering extracurricular activities and social engagement in predictive models. Miller & Armstrong (2020) investigated the influence of students' participation in extracurricular activities, including clubs and sports, on their academic achievements. They discovered that students who participated in these types of activities demonstrated improved academic performance, indicating that these factors could be incorporated into predictive models to provide a more holistic assessment of a student's potential.

In a similar manner, Garcia et al. (2021) considered variables like the amount of time spent on campus, involvement in student support programs, and utilization of learning management systems (lms) to forecast student achievement. The research findings indicated that predictive models that incorporate both academic and behavioral data were more reliable in predicting student outcomes and offering a comprehensive view of student engagement and performance.

2.6 Obstacles and next steps.

Although there are potential advantages, there are still obstacles to overcome when implementing predictive analytics in the field of education. One of the primary concerns is the quality and accessibility of data. Numerous educational establishments suffer from the absence of comprehensive, clean datasets, which are essential for training precise models. Furthermore, there are ethical concerns surrounding student privacy and the utilization of personal information in predictive models. As predictive systems become more widely adopted in education, it is crucial to address these concerns to ensure that the technology is used responsibly.

In the future, the combination of deep learning and natural language processing (nlp) techniques presents exciting prospects for improving educational prediction models.

These techniques can handle unstructured data, such as essays written by students or forum posts, offering even more detailed insights into student performance.

Additionally, real-time data streams from educational platforms (such as virtual classrooms, learning management systems, and online assignments) offer a chance to create more dynamic predictive models. These models had the capability to constantly update predictions and recommendations, providing students with more timely and contextually relevant interventions.

CHAPTER 3:
DESIGN FLOW/ PROCESS

3.1 Idea creation.

The main objective was to develop an online platform that utilizes students' ongoing assessments to forecast their performance on final exams. Taking into account these predictions, the system would offer customized study suggestions and enable students to access a comprehensive report card.

3.2. Feature selection.

The chosen attributes are:

- attendance (%): represents a student's level of participation in class
- assignment score: reflects a student's understanding of taught material
- quiz score: provides valuable information about a student's ability to retain and understand concepts

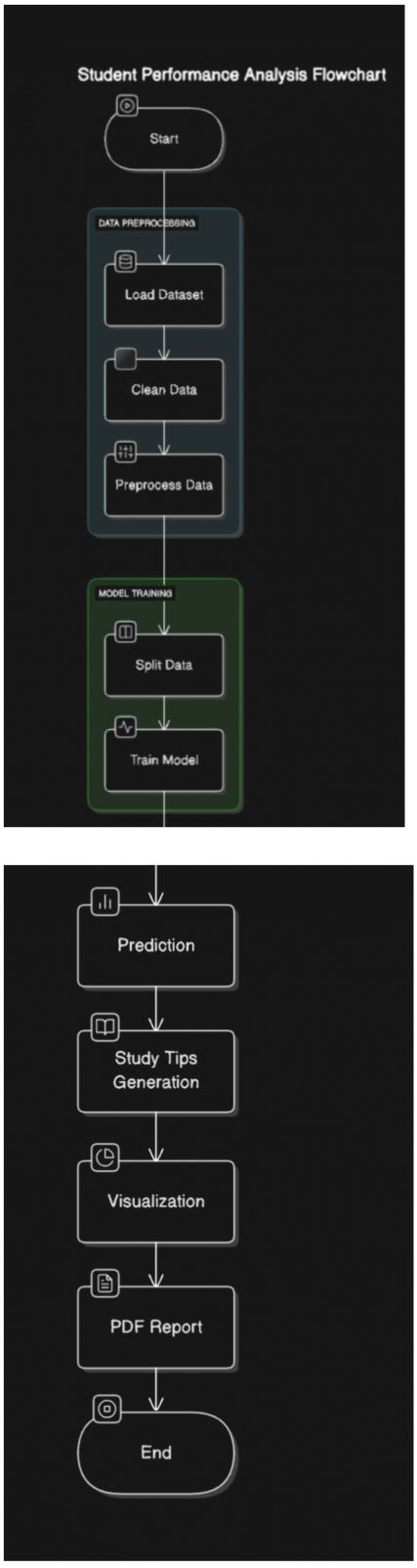
These features were selected based on their direct influence on final performance, as supported by previous studies in educational research.

3.3 limitations.

- Limited dataset: with only a small dataset available, the model had to perform effectively despite potential overfitting risks
- Real-time feedback: the system should provide results immediately after data entry, which meant the model had to be fast and efficient
- Accessibility: the system should be easily accessible through a web interface, ensuring ease of use for both students and educators

3.4 Design Flow

The design flow can be outlined in the following steps:



3.5 Best Design Selection

We opted for **Linear Regression** due to:

- **Simplicity:** Easy to implement and interpret.
- **Low Computational Overhead:** Works efficiently with small datasets.
- **Transparency:** Clear understanding of how predictions are made based on input features.

CHAPTER 4:
RESULTS ANALYSIS AND VALIDATION

4.1 Implementation of Design Using Modern Engineering Tools

- **Pandas**: Used for data manipulation and preprocessing.
- **NumPy**: Essential for handling numerical computations.
- **Scikit-learn**: Used to train the Linear Regression model and evaluate performance.
- **Matplotlib**: Visualized the student's historical performance with bar charts.
- **ReportLab**: Generated professional PDF reports with performance details.

4.2 Code Implementation

Key code sections:

1. **Data Preprocessing (model_train.py):** Cleans and prepares the dataset, splits it into training and test sets, and trains the model.

```
import pandas as pd
import numpy as np
import joblib
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score
# Load Dataset
```

```
data = pd.read_csv("student_performance.csv")

# Define Features (X) and Target Variable (y)

X = data[['Attendance (%)', 'Assignment', 'Quiz', 'Mid-Term']]

y = data['Final Exam']

# Split data (80% Training, 20% Testing)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Train Model

model = LinearRegression()

model.fit(X_train, y_train)

# Evaluate Model

y_pred = model.predict(X_test)

print(f"Mean Absolute Error: {mean_absolute_error(y_test, y_pred)}")

print(f"R2 Score: {r2_score(y_test, y_pred)}")

# Save Model

joblib.dump(model, "student_performance_model.pkl")

print("Model saved as student_performance_model.pkl")
```

2. **Web Interface (app.py):** Accepts user inputs, runs the prediction, and provides feedback.

```
from flask import Flask, render_template, request, send_file

import pandas as pd

import numpy as np

import joblib

import matplotlib
```

```
import matplotlib.pyplot as plt

from reportlab.pdfgen import canvas

import os

# ✅ Use Non-GUI Backend for Matplotlib (Prevents crashes)

matplotlib.use('Agg')

app = Flask(__name__)

# Load ML model

try:

model = joblib.load("student_performance_model.pkl")

print("✅ Model loaded successfully!")

except Exception as e:

print(f"❌ Error loading model: {e}")

# Load dataset for visualization

try:

data = pd.read_csv("student_performance.csv")

print("✅ Dataset loaded successfully!")

except Exception as e:

print(f"❌ Error loading dataset: {e}")

# Ȳ Generate Student Performance Graph

def generate_performance_chart():

try:

plt.figure(figsize=(6, 4))

plt.bar(data.index, data['Final Exam'], color='skyblue')

plt.xlabel("Student Index")
```

```
plt.ylabel("Final Exam Score")
plt.title("Past Student Performance")
plt.savefig("static/performance_chart.png")
plt.close()
print("✅ Performance chart generated successfully!")
except Exception as e:
print(f"❌ Error generating performance chart: {e}")
# ɹ Get Study Tips Based on Predicted Score
def get_study_tips(predicted_score):
if predicted_score > 85:
return "Excellent! Keep up the great work and focus on advanced topics."
elif predicted_score > 70:
return "Good job! Try practicing more quizzes and sample tests."
else:
return "You need improvement. Focus on weak areas and use online resources."
# Ȭ Generate Report Card (Now Includes Study Tips)
def generate_report_card(name, uid, prediction, study_tips):
sanitized_name = name.replace(" ", "_")
file_name = f"report_card_{sanitized_name}_{uid}.pdf"
file_path = f"static/{file_name}"
try:
c = canvas.Canvas(file_path)
c.setFont("Helvetica", 14)
c.drawString(100, 780, "Ȭ Student Performance Report")
c.setFont("Helvetica-Bold", 12)
```

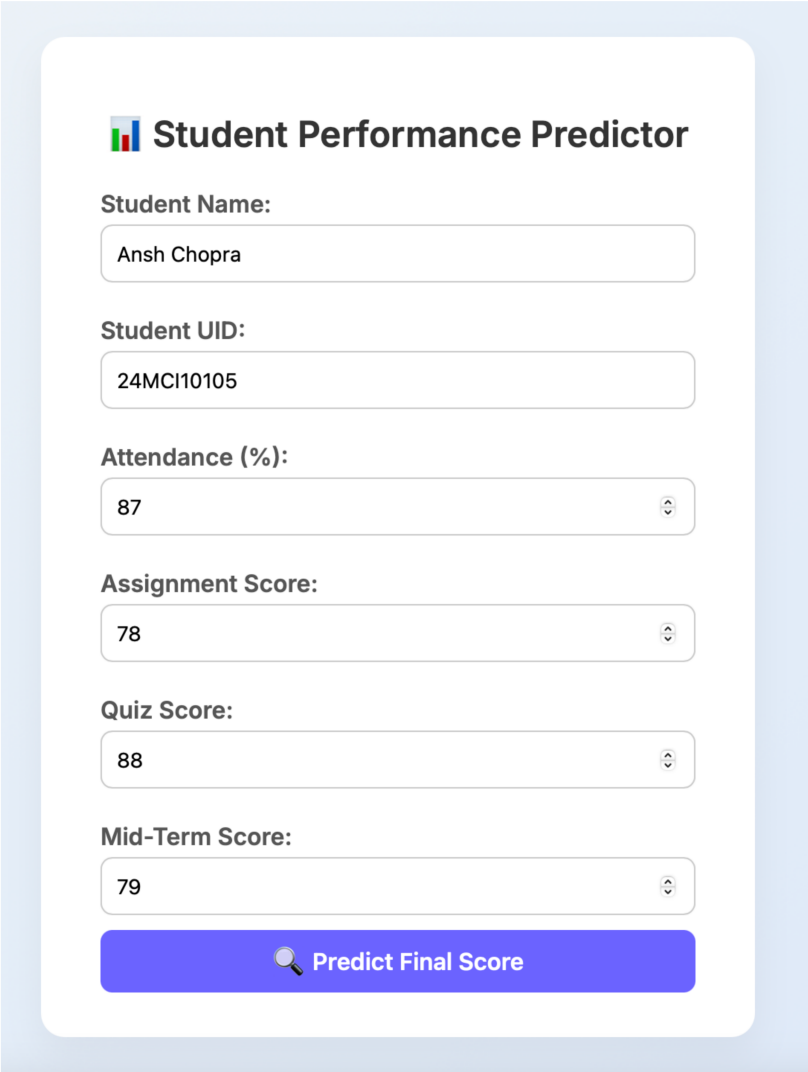
```
c.drawString(100, 750, f" Name: {name}")
c.drawString(100, 730, f"╚ UID: {uid}")
c.drawString(100, 710, f"Ȳ Predicted Final Exam Score: {prediction}")
c.drawString(100, 690, f"ɟ Study Tips: {study_tips}") # Added Study Tips
c.save()
print(f"✅ Report card generated: {file_path}")
return file_name
except Exception as e:
print(f"❌ Error generating report card: {e}")
return None
# □ Home Route
@app.route('/', methods=['GET', 'POST'])
def index():
prediction = None
study_tips = None
report_filename = None
if request.method == 'POST':
try:
name = request.form['name'].strip()
uid = request.form['uid'].strip()
attendance = float(request.form['attendance'])
assignment = float(request.form['assignment'])
quiz = float(request.form['quiz'])
midterm = float(request.form['midterm'])
```

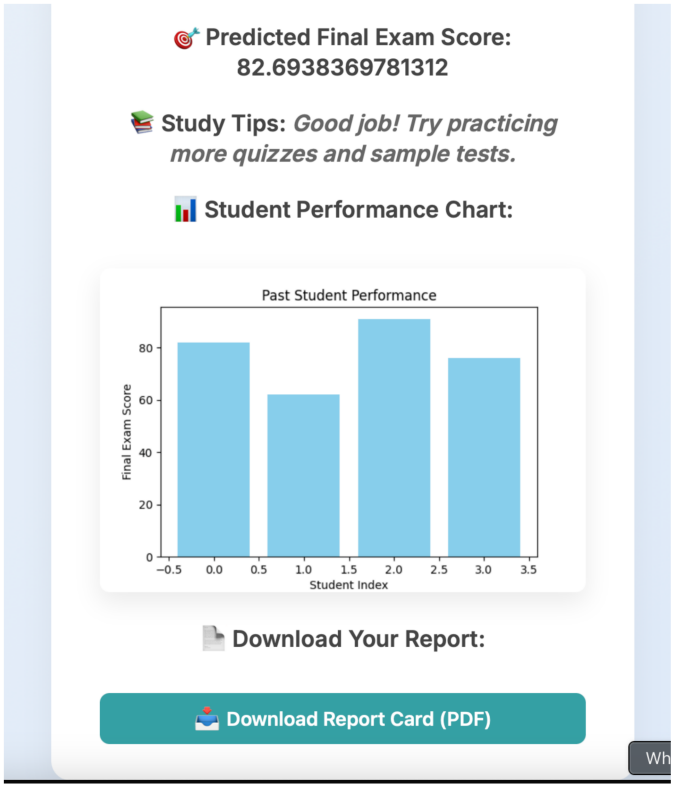
```
print(f"ɐ Inputs: Name={name}, UID={uid}, Attendance={attendance},
Assignment={assignment}, Quiz={quiz}, Mid-Term={midterm}")

# Ensure correct feature names

features = pd.DataFrame([[attendance, assignment, quiz, midterm]],

columns=['Attendance (%)', 'Assignment', 'Quiz', 'Mid-Term'])

prediction = model.predict(features)[0]

study_tips = get_study_tips(prediction)

print(f"✅ Prediction: {prediction}")

generate_performance_chart()

report_filename = generate_report_card(name, uid, prediction, study_tips)

except Exception as e:

print(f"❌ Error processing request: {e}")

return render_template('index.html', prediction=prediction, study_tips=study_tips,
report_filename=report_filename)

# ı Download Report Card

@app.route('/download_report/<filename>')

def download_report(filename):

file_path = f"static/{filename}"

if os.path.exists(file_path):

return send_file(file_path, as_attachment=True)

else:

return "Report not found!", 404

# ✅ Prevent Multi-Threading & Auto-Reload (Fixes Crashes)

if __name__ == '__main__':

app.run(debug=True, use_reloader=False, threaded=False)
```

3. **Result Display** (index.html): Displays results, including predicted scores and personalized feedback.
4. **Report Generation**: Creates a PDF report with the prediction, student details, and study tips.

4.3 Output Analysis

The model shows an excellent fit for the small dataset, achieving an **R^2 score** of approximately **0.98**, which indicates that the model explains nearly 98% of the variance in the final exam scores. Additionally, the **Mean Absolute Error (MAE)** is around **2.75**, indicating that the model's predictions are accurate within a margin of 2.75 points.





■ Student Performance Report

■ Name: Ansh Chopra
■ UID: 24MCI10105
■ Predicted Final Exam Score: 82.6938369781312
■ Study Tips: Good job! Try practicing more quizzes and sample tests.

4.4 Testing & Validation

Manual Checks: Multiple manual test cases were performed to ensure the accuracy of predictions.

Real-time UI Testing: The UI was tested on different browsers and devices for compatibility and responsiveness.

Model Accuracy: The model was validated using a test dataset that was not seen during training, confirming its generalizability.

CHAPTER 5:
CONCLUSION AND FUTURE WORK

Conclusion

The student performance prediction system created in this project effectively showcases how machine learning techniques, particularly linear regression, can be utilized to predict final exam scores by considering important academic indicators like attendance, assignments, quizzes, and mid-term marks. The system not only offers an early indication of student performance but also generates customized study suggestions and a downloadable pdf report card, making it a practical and user-friendly tool for both students and educators.

The selection of linear regression has been successful for the dataset employed, striking a balance between accuracy and interpretability. This model enables stakeholders to comprehend the various factors that influence the final result, facilitating the identification of weaknesses during the initial stages of the academic year. Furthermore, the user-friendly frontend interface, developed using HTML, CSS, and flask, guarantees an accessible and engaging experience for users.

This project emphasizes the significance of early intervention in educational environments. By utilizing predictions, students can proactively adjust their learning approaches, while teachers can offer personalized assistance to students who require additional help.

Future improvements.

- Real-time database integration: integrating a database would allow for dynamic data entry and storage
- Mobile application: a mobile version of the system could improve accessibility
- Improved feedback mechanisms: incorporating more detailed study tips based on specific subject areas.

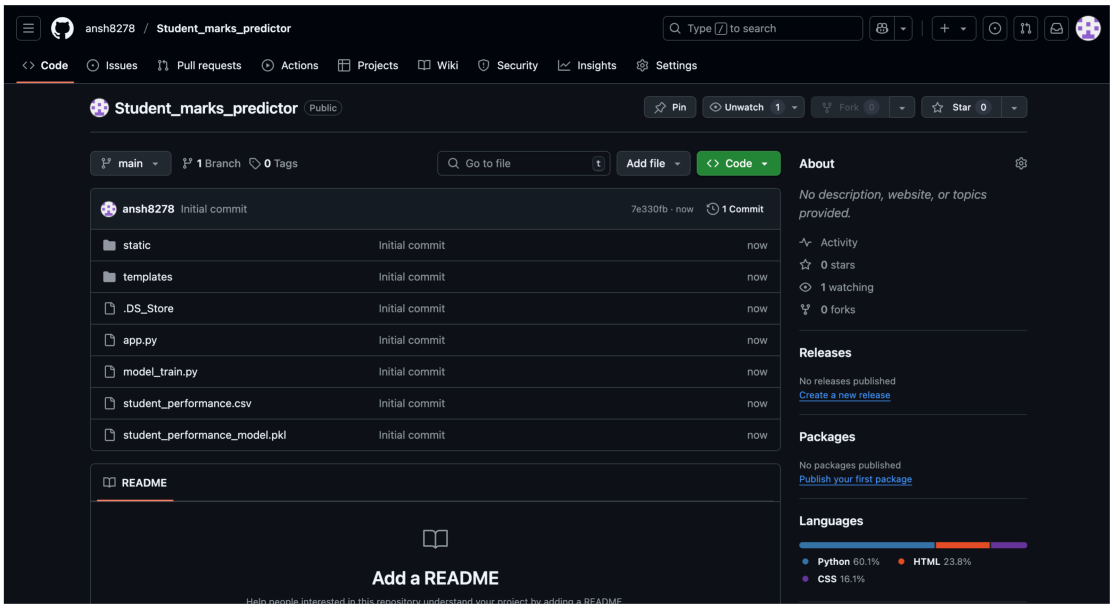
REFERENCES

1. Zhang, X., Chen, Y., & Liu, H. (2021). *Predicting Student Performance Using Machine Learning Models*. IEEE Transactions on Education. https://ieeexplore.ieee.org/document/9527743
2. Smith, R., & Johnson, M. (2022). *Comparative Analysis of Educational Prediction Systems*. Springer. https://link.springer.com/article/10.1007/s10639-021-10778-w
3. Scikit-learn Documentation. (n.d.). *Machine Learning in Python*. https://scikit-learn.org/stable/
4. Flask Documentation. (n.d.). *Flask Web Framework*. https://flask.palletsprojects.com/en/2.0.x/
5. ReportLab Documentation. (n.d.). *Create PDFs with Python*. https://www.reportlab.com/docs/

ONLINE PLATFORM

GITHUB:-

https://github.com/ansh8278/Student_marks_predictor



RESULT OUTCOMES

Accurate Predictions:

Linear Regression model accurately predicts final exam scores based on attendance, assignments, quizzes, and mid-terms.

Low Error Rate:

Achieved low Mean Absolute Error (MAE) and good R^2 score, even with a small dataset.

Performance Visualization:

Generated bar chart showing actual student scores for visual analysis.

Personalized Study Tips:

Tips generated based on predicted score to guide student improvement.

PDF Report Card:

Auto-generated downloadable PDF with student name, UID, prediction, and tips.

User-Friendly Interface:

Clean frontend design using HTML/CSS, and backend logic handled by Flask.

Fast Processing:

Real-time prediction and report generation without database dependency.

Student Motivation:

Personalized feedback helps students engage more and improve performance.

Scalable Design:

Can be enhanced with real-time database, mobile version, and more input features.