STUDENT PERFORMANCE ANALYSIS USING MACHINE LEARNING

Project report submitted to MAHATMA GANDHI UNIVERSITY in partial fulfilment of the requirement for the award of Degree of

BACHELOR OF COMPUTER APPLICATION

Submitted by

SOURABH.S

Reg. No 220021084249

Under the Guidance of

Mrs. HIMA S (Asst. Prof)

DEPARTMENT OF COMPUTER APPLICATION

SETH RAM BAHADUR SINGH GUJARATI COLLEGE, COCHIN

MAHATMA GANDHI UNIVERSITY

MARCH 2025

SETH RAM BAHADUR SINGH GUJARATI COLLEGE

(Approved by AICTE and Affiliated to MAHATMA GANDHI UNIVERSITY, KOTTAYAM)

CERTIFICATE

Certified that the seminar entitled **STUDENT PERFORMANCE ANALYSIS USING MACHINE LEARNING is** a record of the Bonafide work done by **SOURABH S (Reg No:220021084249) under** our guidance and supervision, in partial fulfilment of one of the requirements for the award of the Degree of **Bachelor of Computer Application** from the Mahatma Gandhi University during the year 2024 - 2025.

PROJECT GUIDE **HOD** **PRINCIPAL**

Submitted for the Viva – Voce Examination held on .....

INTERNAL EXAMINER EXTERNAL EXAMINER

College Seal

An ISO 9001 : 2015 Certified Organization

LCC
Computer Education

March 14, 2025

TO WHOM SO EVER IT MAY CONCERN

This is to certify that **SOURABH S** 6th Semester **BCA** student of **SETH RAM BAHADUR SINGH GUJARATI COLLEGE** has successfully completed a project titled "**STUDENT PERFORMANCE ANALYSIS USING MACHINE LEARNING**" from our organization.

The duration of the project was for 3 months. The Project was incorporated in **PYTHON** and was implemented successfully.

Thanking you,

For **LCC Computer Education**

Ramaswamy
Director

LCC COMPUTER EDUCATION LCC COCHIN - 31

DECLARATION

I hereby declare that the Seminar entitled STUDENT PERFROMANCE ANALYSIS USING MACHINE LEARNING submitted to the Mahatma Gandhi University in partial fulfilment of one of the requirements for the award of the Bachelor of Computer Application is a record of original work done by me during DECEMBER 2024 – MARCH 2025 under the supervision and guidance of Mrs. Hima S ,Assistant Professor, Department of Computer Application, Seth Ram Bahadur Singh Gujarati College, Cochin and this is an original work which has not been submitted previously for the award of any degree or diploma to this University or any other University.

PLACE:

DATE:

Name and Signature of Candidate

ACKNOWLEDGEMENT

"Salutations to the Guru, the supreme power above who provide us knowledge and save us from all pangs of ignorance" Mother Nature has endowed us with a bounty of resources from which we always borrow; My thanks for thy blessings".

I express my deep sense of gratitude to the College management, Seth Ram Bahadur Singh Gujarati College, Kochi for given me the opportunity to undertake this project work. I would like to express my sincere gratitude to my Project Guide, **Mrs. HIMA S**, Assistant Professor, and Head of the department **Mrs. Vidhya R**, for their continuous support in my work, study and for their endurance, inspiration and immense knowledge. Their Guidance helped me in all the time of my project work. I could not have imagined having a better advisor and mentor for my project. My sincere thanks to **Dr. Sherly Prakash**, Principal Seth Ram Bahadur Singh Gujarati College, Kochi, for providing me the necessary facilities in completing the Project work successfully. My sincere thanks to My Family Members and Friends for their encouragement and support to complete my project successfully. My heartfelt thanks to all.

My heartfelt thanks all

SOURABH S

TABLE OF CONTENTS

1. **CHAPTER 1: INTRODUCTION**

i. Abstract i

ii. Institute Profile.....ii

1. Introduction 2

1.1 Objectives..... 3

1.2 Need for Project 3

2. **CHAPTER 2: SYSTEM ANALYSIS**

2.1 Existing System.....6

2.2 Proposed System 7

2.3 Feasibility System 8

3. **CHAPTER 3: SYSTEM REQUIREMENT**

3.1 Hardware Requirements 11

3.2 Software Requirements 11

4. **CHAPTER 4: SYSTEM DESIGN**

4.1 Input Forms 13

4.2 Architecture Design..... 14

4.3 Context Level..... 15

4.4 Processes 17

5. **CHAPTER 5: DATA COLLECTION and PREPROCESSING .....** 20

6. **CHAPTER 6: MODEL SELECTION and EVALUATION**

6.1 Model Selection.....24

6.2 Selected Model..... 24

6.3 Model Performance Results 24

7. **CHAPTER 7: LOGISTIC REGRESSION**

7.1 Logistic Regression Function 27

7.2 Importance 27

8. **CHAPTER 8: NORMALIZATION**

8.1 Database Normalization 29

8.2 Feature Normalization for Machine Learning 30

9. **CHAPTER 9: IMPLEMENTATION**

9.1 Technologies Used 33

9.2 Machine Learning Model 33

9.3 Django Integration 34

9.4 User Interface 35

10. **CHAPTER 10: SYSTEM TESTING**

10.1 Software Testing 37

10.2 ML Model Testing and Evaluation 38

10.3 Performance Testing 39

11. **CHAPTER 11: SYSTEM IMPLEMENTATION**

11.1 Steps in System Implementation 41

11.2 Steps in ML Model Implementation 42

12. **CHAPTER 12 : INSIGHTS GAINED** 44

13. **CHAPTER 13: CONCLUSION** 48

14. **CHAPTER 14: BIBILOGRAPHY** 51

15. **CHAPTER 15: APPENDIX** 54

16. **CHAPTER 16: SOURCE CODE** 59

CHAPTER 1

INTRODUCTION

ABSTRACT

In the field of education, evaluating student performance is essential for ensuring academic success and early intervention. This project, Student Performance Analysis Using Machine Learning, utilizes a Logistic Regression model to predict student performance based on core subject scores. The model processes input features such as math score, reading score, and writing score, applying data preprocessing techniques including categorical encoding, feature scaling, and dataset partitioning to ensure accuracy. The system is built using Django for a web-based interface, where students can input their scores and receive real-time feedback on their predicted performance. The model classifies students into two categories: High Performer or Needs Improvement, assisting educators in identifying at-risk students and implementing necessary interventions. By leveraging machine learning techniques like label encoding, data normalization, and logistic regression, this project bridges the gap between academic data analysis and real-world decision-making. It provides an effective and scalable approach for schools and educational institutions to enhance student success and resource allocation.

ABOUT US

Founded in Cochin back in the year 1992, LCC started its journey as a premier computer coaching centre and later advanced into one of the most reputed brands for computer learning throughout India. LCC has surpassed victory in finding and refining those hidden talents in our pupils, to deliver thousands of IT presto the evergreen IT Industry every year. Whether you are a novice, a mediocre or a professional in computer technologies, we have the right course for you to advance and follow an ambitious career path.

Technology as you can imagine, advances every second. It feeds upon itself i.e., technology makes more technology possible. LCC's expert academicians and certified faculties interact with our leading computer technology company allies like(include LCCs alliances here for eg:Java, Red Hat, Microsoft and Oracle)to come up with relevant certification courses (short term, long term, professional, and career courses) after thorough market research and study is also the training partner to World famous MNC TOTAL SOLUTIONS INC, CHICAGO, USA, and its allied Company EZMRX Bangalore. Our curriculum features the largest offering of computer course varieties to keep you updated with cutting edge technology through expert and experienced training faculties. This will help the students pursue a job of their dreams with better career prospects and industry best salaries.

That is not all to it; we also mould students into professional characters for interviews through job fairs, placement workshops, seminars, presentations and various other personality development programs. To sum it up, in spite of being the right platform to build your sharp IT knowledge, LCC also supports your first step to career by introducing you to the complex technology job sector through placement opportunities to reputed companies.

Come join us and innovate to multiply world's Technology by your skill, who knows, tomorrow we might get a chance to teach your invention!

LIVE PROJECTS

LCC offers advanced internship projects to its students by letting them work on live projects using cutting edge languages and tools like ASP.net, C#, Java and J2EE, J2ME, & PHP. Our aim is to mould the overall skillset and personality of our students to make them rise up to the current industry expectations thereby improving their employability levels. We makes it possible by letting them learn under and Interact with the top developers and crews of our leading tech company allies, thus ensuring excellent exposure to real time work environment and practises. The training helps reduce the gap between the learning processes at colleges and real time working environments at companies.

BTech, BE, MSc, BSc, MCA, BCA and Polytechnic students are provided training with our leading IT company allies such as Suyati Technologies, Ojus Healthcare, Young Soft and otherprominentcompanies at Info Park, Kochi. Being the training partner to World famous MNC TOTAL SOLUTIONS INC, CHICAGO, USA, and its allied Company EZMRX

Bangalore, LCC has proven its might in offering world class computer knowledge to its pupil through its IT connoisseurs. Our internship program is well recognized by top MNCs who have taken keen interest to recruit from us every year.

CORPORATE TRAINING

Technology evolves on a daily basis. Sometimes it becomes necessary to train even the smartest developer on the latest IT developments, market trends and challenges to keep up your company's staff intellect. We understand these requirements and have a dedicated team to identify the cream of the crop solutions relevant for your staff to excel and grow beyond horizons. Our corporate training sessions have helped start-ups to global enterprises to remain ahead of the technology curve. We have extended our training to major corporate, banking / financial institutions, Central / State Government organizations, and public / private sector companies across the country.

Our corporate training programs cover a lot of areas that are closely connected with employees' professional as well as personal life, aiming at injecting positive energy in their endeavours. This has helped employees to be more productive which has in turn resulted in better ROIs for the company. Over 98% of our clients testify that they have utilized the tips and tricks we provided during sessions. And over 50% ofemployee performance improvements noticed in companieswas directly attributed to our brainstorming sessions for the keen learners.

Training programs are conducted either at clients' place or at the LCC headquarters in Cochin. We carefully evaluate the clients' requirements and advice programs and packages relevant to their needs. Training programs encompass almost everything from the very basic to the latest and most advanced techniques and practices, with extensive hands-on and instructor-led sessions.

OUR CAPABILITIES

LCC's advisory committee of IT professionals studies the hot market trends consulting with leading computer technology providers in order to revise the offered courses/subjects every year. Curricula are then updated to reflect current and future technology drifts and demands. LCCs computer training program splits into subsequent four major sectors:

LCC
Computer Education

EDUCATION
SOFTWARE TRAINING

LIAD
MULTIMEDIA TRAINING

SOFT SOLUTIONS
SOFTWARE DEVELOPMENT

INTRODUCTION

In the modern education system, assessing student performance is crucial for ensuring academic success. Traditionally, institutions have relied on manual grading and subjective evaluations, which can be **time-consuming, prone to errors, and inefficient for large student populations**. With the rise of **data-driven technologies**, machine learning provides a powerful approach to analyzing student performance more accurately and effectively.

The primary objective of this project is to develop an **intelligent system that predicts and analyzes student performance** based on various academic and demographic factors. By employing **machine learning algorithms**, the system identifies patterns, correlations, and trends in student data, enabling educators to make **informed, data-driven decisions**. These insights can assist in designing **personalized learning plans**, identifying **at-risk students** early, and improving overall institutional effectiveness.

The proposed system seeks to overcome the limitations of **existing manual and semi-automated evaluation methods**, which often struggle with scalability and accuracy. **By introducing an automated, AI-powered solution**, this project addresses the growing need for **efficient, intelligent academic evaluation techniques**. It leverages **Logistic Regression** for performance classification and integrates seamlessly with a **Django-based web interface**, making it accessible and user-friendly for both students and administrators.

In today's **data-centric world**, the ability to transform raw academic records into **meaningful insights** holds immense potential for driving student success. By analyzing key factors such as **math, reading, and writing scores**, this project aims to predict performance outcomes while providing actionable recommendations for improvement.

From tackling **academic stress and dropout risks** to enhancing overall learning outcomes, this project envisions a **holistic approach to performance analysis**. Through a combination of **cutting-edge machine learning techniques, intuitive data visualization, and user-centric design**, this system is poised to make a **meaningful impact on the education sector**. As educational institutions strive for excellence, this project offers a **forward-thinking solution** to ensure that **no student is left behind in their academic journey**.

OBJECTIVES

- To develop a machine learning model capable of predicting student performance based on input features.
- To integrate the model into a Django-based web application.
- To provide an admin panel for managing students and viewing performance results.
- To offer a user-friendly interface for students to check their predicted performance.

NEED FOR THE PROJECT

In today's educational ecosystem, monitoring and enhancing the performance of students is an essential function for learning institutions. Conventional evaluation schemes, like manual evaluations and subjective judgments, tend to be unscalable, inefficient, and inaccurate when faced with large numbers of students. As technology and data-driven tools become prevalent, there is a compelling need for an intelligent, automated, and scalable process in evaluating student performance.

- Early Student At-Risk Identification

Machine learning facilitates predictive analysis to identify struggling students even before their grades significantly deteriorate. This aids institutions in extending early interventions to avert failures.

- Data-Driven Decision Making

Conventional methods of assessing student performance depend on manual analysis of student records, which can be susceptible to prejudice. A machine learning-based approach guarantees prejudice-free, data-driven predictions that support educators in making sound decisions.

- Personalized Learning Experiences

Through performance trends analysis, the system assists in creating personalized learning pathways, which are tailored to students according to their strengths and weaknesses.

- Optimal Resource Utilization: Institutional players can leverage predictive insights to optimize resources, including tutoring programs, mentorship, and curriculum alignment, to ensure optimal utilization of institutional support.

- Automation & Scalability

In contrast to manual assessment procedures, an AI-driven solution enables universities and schools to examine thousands of student records in real-time, which is a scalable process ideal for big institutions.

- Minimizing Dropout Rates & Maximizing Student Success

Predictive analytics enable institutions to identify patterns in academic distress, falling grades, and at-risk dropout, making it possible to intervene early and improve student retention.

- Maximizing Institutional Effectiveness

By using performance evaluation automation and data visualization methods, administrators can have a good view of students' overall development, enhancing academic programming and policy-making.

CHAPTER- 2
SYSTEM ANALYSIS

INTRODUCTION TO SYSTEM ANALYSIS

System analysis is the process of studying, evaluating, and designing a system to ensure it meets user requirements efficiently. It involves breaking down a system into its components to understand its functionality, identify improvements, and optimize performance.

The primary goal of system analysis is to **identify user needs, define system requirements, and ensure smooth implementation**. This process helps in eliminating inefficiencies, reducing errors, and enhancing system reliability. System analysis typically includes **requirement gathering, feasibility study, data analysis, and system modeling**. It also plays a critical role in selecting the right technologies, defining workflows, and ensuring that the final system aligns with organizational goals.

By following a structured system analysis approach, developers can create **efficient, scalable, and user-friendly solutions** that improve productivity and decision-making across various domains.

EXISTING SYSTEM

The traditional methods of analyzing student performance rely on manual assessment, report card evaluations, and teacher observations. These methods, while commonly used, have several limitations:

1. Limitations of the Existing System:

- Manual and Time-Consuming: Evaluating student performance based on exam results and teacher feedback takes a lot of time and effort.
- Lack of Predictive Insights: Traditional systems only assess past performance but do not predict future academic success or identify at-risk students early.
- Subjectivity and Human Error: Teacher evaluations can be subjective, leading to inconsistencies in grading and performance assessment.
- Limited Data Utilization: Many institutions do not use available student data effectively to derive insights and improve learning outcomes.
- No Real-Time Feedback: Students and educators do not receive immediate insights, making it difficult to take timely corrective actions.

2. Challenges Faced in the Existing System:

- Inability to handle large datasets efficiently.
- Lack of automation, requiring manual record-keeping.
- Difficulty in tracking long-term academic progress.
- No personalized learning plans based on student performance trends.

3. Need for an Improved System:

Due to these limitations, there is a strong need for a data-driven, automated, and intelligent system that can analyze student performance efficiently, provide predictive insights, and assist in early interventions. Machine learning-based solutions can bridge these gaps by leveraging historical student data to identify patterns, predict outcomes, and suggest personalized improvements.

This project aims to overcome the challenges of the existing system by providing an automated, accurate, and scalable solution using machine learning and web-based technologies.

Proposed System

The Proposed System aims to overcome the limitations of traditional student performance evaluation methods by leveraging Machine Learning (ML) and Web-Based Technologies. This system provides an automated, accurate, and scalable approach to analyze and predict student performance, helping educators make data-driven decisions.

1. Features of the Proposed System

- Machine Learning-Based Performance Prediction:
    - Uses Logistic Regression to classify students as "High Performer" or "Needs Improvement" based on their academic scores.
- Web-Based Platform (Django Framework):
    - Provides an interactive user interface for students and administrators to input, analyze, and manage student performance data.
- Automated Data Processing: Preprocesses student academic records, applies data normalization, and performs real-time analysis.

- Early Identification of At-Risk Students:
    - Helps educators intervene at an early stage and implement personalized learning strategies.
- Admin Dashboard for Institutional Management:
    - Enables administrators to manage student records, view predictions, and analyze performance trends.
- Secure and Scalable Database (SQLite):
    - Stores student information efficiently, ensuring data integrity and security**.**

➢ FEASIBILITY STUDY

A **feasibility study** is conducted to evaluate whether the proposed system is practical, cost-effective, and beneficial. This study ensures that the **Student Performance Analysis Using Machine Learning** project is viable before full-scale implementation. The feasibility study includes **technical, economic, operational, and legal aspects** to assess the system's success.

Technical Feasibility

1. **Technology Stack:** The system uses **Django (Python)** for the web application, **Scikit-Learn** for machine learning, and **SQLite** for database management. These technologies are well-supported and widely used.
2. **Machine Learning Implementation:** The use of **Logistic Regression** ensures that the prediction model is efficient and requires minimal computational resources.
3. **Scalability:** The system can handle large datasets and can be upgraded with more advanced ML models if needed.

➢ ECONOMIC FEASIBILITY

The economic feasibility of this project assesses whether the Student Performance Analysis Using Machine Learning system is cost-effective and delivers value to educational institutions. The goal is to ensure that the system can be developed, implemented, and maintained within a reasonable budget while providing long-term benefits such as improved student performance monitoring and early intervention.

1. **Cost-Effective:** The system is built using open-source technologies, reducing development costs. The project is developed using **open-source technologies** such as **Django (Python), Scikit-Learn, and SQLite**, minimizing software costs.
2. **Minimal Hardware Requirements:** Since the model is lightweight, it does not require high-end hardware, reducing infrastructure expenses.
3. **Long-Term Benefits:** The system helps in early student intervention, improving institutional efficiency and reducing dropout rates. The system enhances **student performance tracking**, leading to **better academic outcomes** and institutional efficiency.
4. **Early Intervention –** Helps identify at-risk students early, reducing dropout rates and improving student success.

OPERATIONAL FEASIBILITY

The purpose of the operational feasibility is to determine whether the new system will be used if it is developed and implemented and whether there will be resistance from users that will undermine the possible application benefits .The aspect of study is to check the level of acceptance of the system by the user .This includes the process of training the user to use the system efficiently .The user must not feel threatened by the system, instead must accept it as a necessity .The level of acceptance by the user solely depends on the methods that are employed to educate the user about the system and to make him familiar with it .This level of confidence must be raised so that he is also able to make some constructive . The proposed system is an upgrade version of the current systems new fields have been implemented according to the user need, hence it ensures all the aspects.

Operational feasibility evaluates whether the Student Performance Analysis Using Machine Learning system can be successfully integrated into educational institutions and used effectively by students, teachers, and administrators. It ensures that the system meets user needs, improves efficiency, and is practical to implement in real-world scenarios.

- User-Friendly Interface
- Automation of Student Performance Analysis
- Accuracy & Reliability

CHAPTER-3

SYSTEM REQUIREMENTS

The system requirements define the **hardware, software, and functional specifications** necessary for successfully implementing and running the **Student Performance Analysis Using Machine Learning** system.

HARDWARE REQUIREMENTS

Category	Requirement
Development Environment	
Processor	Intel Core i3 or higher / AMD equivalent
RAM	Minimum 4GB (Recommended: 8GB for better performance)
Storage	At least 20GB of free space
Operating System	Windows, macOS, or Linux
Graphics Card	Not required (as the project does not involve deep learning)
Deployment (Server Requirements)	
Processor	Intel Core i5 or higher (or equivalent cloud-based CPU)
RAM	Minimum 8GB (for handling multiple user requests)
Storage	50GB+ (for dataset storage and logs)
Database Server	SQLite (default) / MySQL / PostgreSQL

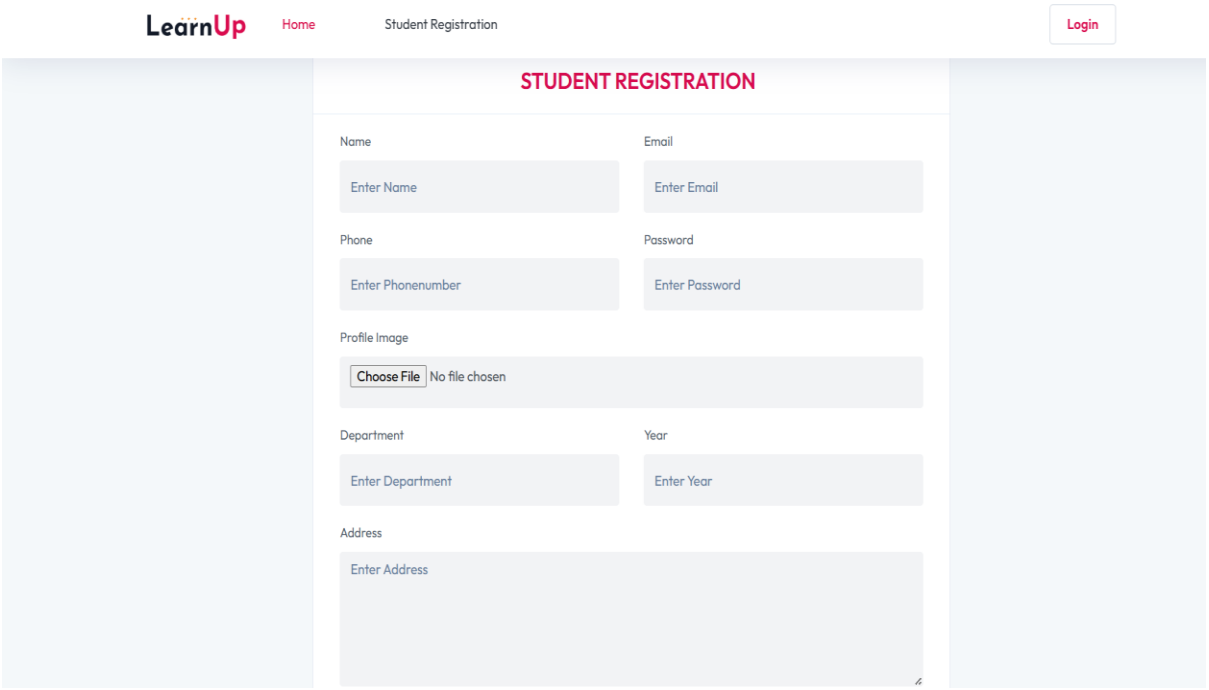
SOFTWARE REQUIREMENTS

Category	Requirement
Operating System	Windows, macOS, or Linux
Programming Language	Python 3.x
Framework	Django (for web application development)
Machine Learning Libraries	Scikit-learn, Pandas, NumPy, Matplotlib/Seaborn
Database	SQLite (default) / MySQL / PostgreSQL
Frontend Technologies	HTML, CSS, Bootstrap (for UI design)
Backend Technologies	Django (Python), REST API (if needed)
IDE/Code Editor	VS Code, PyCharm, or Jupyter Notebook
Version Control	Git & GitHub
Web Server	Apache / Nginx / Gunicorn (for deployment)
Hosting Platform	Local server / Cloud-based (AWS, Azure, or Heroku)

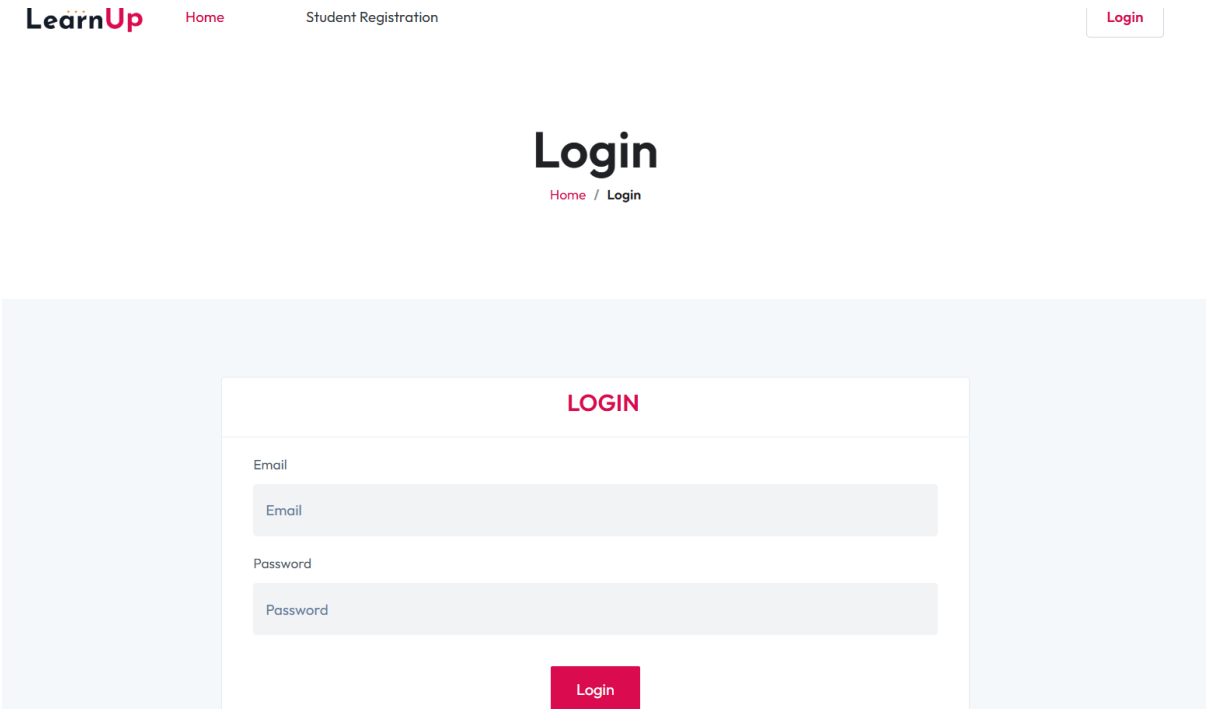
CHAPTER-4
SYSTEM DESIGN

INPUT FORMS

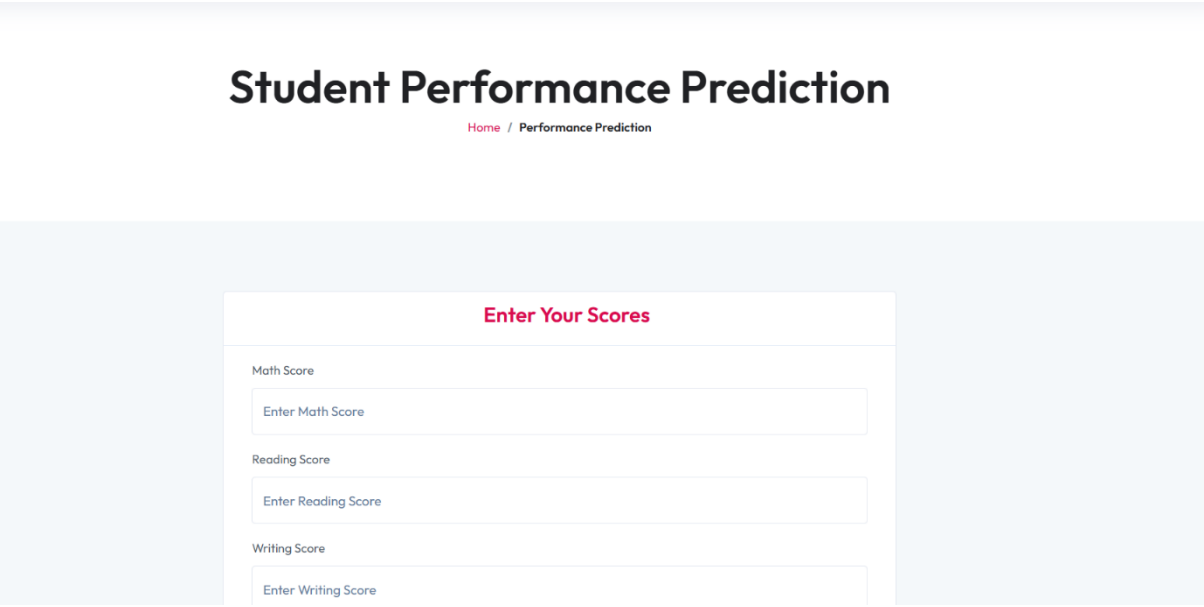
1.1 Student Registration



1.2 Login Form



1.3 Prediction Page



➢ ARCHITECTURE DESIGN

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

It shows how data enters and leaves the system, what changes the information, and where data is stored. The objective of a DFD is to show the scope and boundaries of a system as a whole.

It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a **data flow graph** or **bubble chart.**

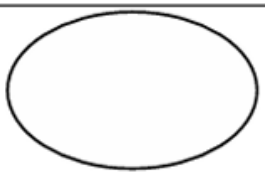
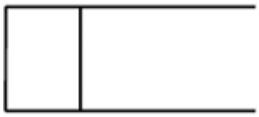
Entities: Entities include source and destination of the data. Entities are represented by rectangle with their corresponding names**.**

Process: The tasks performed on the data is known as process. Process is represented by circle. Somewhere round edge rectangles are also used to represent process**.**

Data Flow: The movement of data in the system is known as data flow. It is represented with the help of arrow**.** The tail of the arrow is source and the head of the arrow is destination**.**

Data Store: Data storage includes the database of the system. It is represented by rectangle with both smaller sides missing or in other words within two parallel lines.

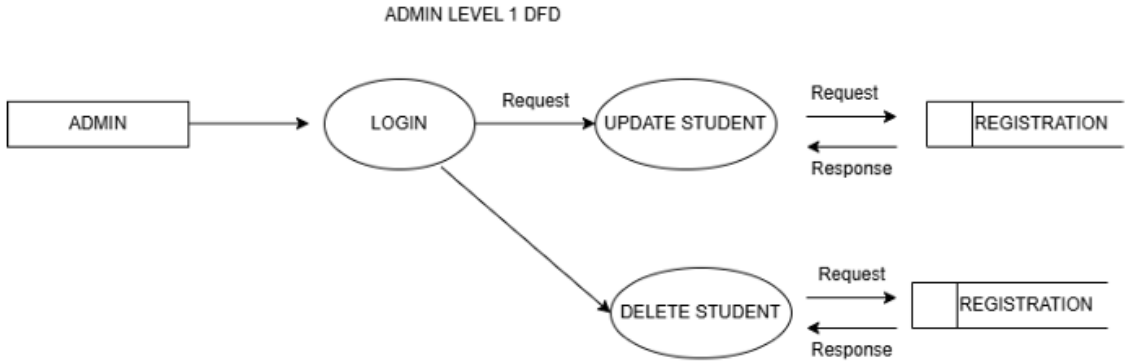
The following are some DFD symbols used in the project:

SYMBOL	NAME	MEANING
	Entity	It represents an external entity
	Process	It represents a process
	Data Flow	It represents data flow
	Data Store	It represents data store

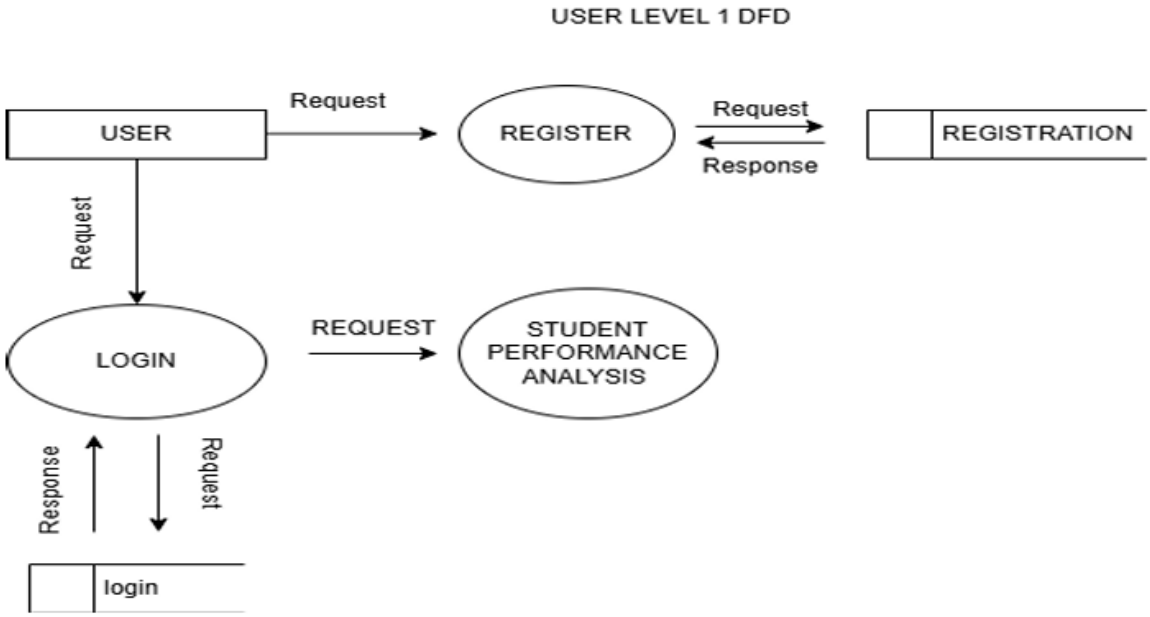
CONTEXT LEVEL

LEVEL 0

LEVEL 1 ADMIN



LEVEL 1 USER



PROCESSES

1. **Entities**

• Student: Registers, logs in, inputs data, checks performance.

• Admin: Logs in, manages students (view/update/delete). • System: Processes data and stores information.

2. **Data Flow**

1. Student provides login credentials → System authenticates user.

2. Student enters performance data → System processes ML prediction.

3. Admin logs in → System verifies credentials.

4. Admin views student list → System retrieves data from the database.

5. Admin updates/deletes student records → System updates the database.

2.1 DFD Level 1 (Detailed Flow) Processes

1. Student Registration & Login → Stores credentials in User Database.

2. Student Inputs Performance Data → Sent to ML Model for processing.

3. System Processes ML Prediction

4. Admin Logs In → System authenticates credentials using User Database.

5. Admin Manages Students (View/Update/Delete) → Modifications are reflected in User Database.

Field Name (Key)	Full Form	Data Type	Description
id	ID (Primary Key)	AutoField (Primary Key)	Unique identifier for each student (auto-generated).
logid	Login ID (Foreign Key)	ForeignKey (User)	Links to the **User** model for authentication.
username	Username	CharField (max_length=20)	Stores the student's chosen username.
email	Email Address	EmailField (Unique)	Stores the student's email (ensures uniqueness).

phone_number	Phone Number	CharField (max_length=15)	Stores the student's contact number, allowing country codes.
image	Residential Address	ImageField (upload_to="images/")	Stores an optional profile picture..
address	Residential Address	CharField (max_length=200, null=True, blank=True)	Stores the student's home address.
dept	Department	CharField (max_length=100)	Stores the student's department (e.g., Computer Science).
Year	Year of Study	CharField	Stores the student's current year in college.
status	Registration Status	CharField (Choices: Pending, Approved, Rejected)	Stores the approval status of the student.

LOGIN TABLE

Field Name (Key)	Full Form	Data Type	Description
id	ID (Primary Key)	AutoField (Primary Key)	Unique identifier for each user (auto-generated).
username	Username (Inherited)	CharField (Unique)	Stores the login username (inherited from AbstractUser).
password	Password (Inherited)	CharField	Stores the user's password securely (inherited).
email	Email Address (Inherited)	EmailField (Unique)	Stores the user's email (inherited).
usertype	User Type	CharField (max_length=20)	Defines the type of user (e.g., Admin, Student, Teacher).
viewPassword	Viewable Password (Unsafe)	CharField (max_length=200, null=True)	Stores password in plain text (**Security Risk**).

PERFORMANCE TABLE

Field Name	Data Type	Description
`id` (PK)	AutoField	Unique record ID
`student_id` (FK)	ForeignKey(Student_reg)	Links to Student table
`math_score`	FloatField	Student's math marks
`reading_score`	FloatField	Student's reading marks
`writing_score`	FloatField	Student's writing marks
`average_score`	FloatField	(math + reading + writing) / 3
`performance_status`	CharField(20)	High Performer / Needs Improvement

CHAPTER-5
DATA COLLECTION and PREPROCESSING

Introduction

Data collection and preprocessing are essential steps in building a machine learning model, ensuring the dataset is clean, relevant, and structured for optimal performance. In the Student Performance Analysis project, these steps help in analyzing academic performance and predicting outcomes efficiently.

2. Data Collection

Data is collected from **educational records** or publicly available datasets such as the **StudentsPerformance.csv** dataset. This dataset includes information about students' scores in core subjects and other relevant attributes affecting academic performance.

Sources of Data:

- Student academic records (Math, Reading, and Writing scores)
- Demographic details (Gender, Ethnicity, Parental Education Level)
- Learning environment factors (Lunch Type, Test Preparation Course)

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75
...	...	...	...	...	...	...	...	...
995	female	group E	master's degree	standard	completed	88	99	95
996	male	group C	high school	free/reduced	none	62	55	55
997	female	group C	high school	free/reduced	completed	59	71	65
998	female	group D	some college	standard	completed	68	78	77
999	female	group D	some college	free/reduced	none	77	86	86

1000 rows × 8 columns

3. Data Preprocessing

Before training the machine learning model, raw data must be cleaned, transformed, and structured for efficient learning.

Steps in Data Preprocessing:

Handling Missing Values:

- Check for missing data in the dataset and fill/remove them accordingly.

- Example: If any student's score is missing, either replace it with the average or remove the entry.

Feature Selection:

- Selecting only relevant features like math, reading, and writing scores for performance prediction.

Encoding Categorical Data:

- Non-numeric features (e.g., Gender, Race/Ethnicity, Parental Education, Lunch) are converted to numeric values using Label Encoding.

Feature Scaling (Standardization):

- Since math, reading, and writing scores have different scales, we **standardize** them to improve model performance.

```
if request.method == "POST":
    # Collect input from the student
    try:
        math_score = float(request.POST['math_score'])
        reading_score = float(request.POST['reading_score'])
        writing_score = float(request.POST['writing_score'])
    except ValueError:
        return render(request, 'index.html', {'error': 'Invalid score input.'})
```

Splitting Dataset for Training & Testing:

- Data is divided into training (70%) and testing (30%) sets using train_test_split.

```
# Features and Target
X = df[['math score', 'reading score', 'writing score']]
y = (df['average_score'] >= 75).astype(int)

# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Standardize the data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Importance of Data Preprocessing

- Increases Model Accuracy: Helps the model learn patterns efficiently.
- Reduces Bias & Errors: Eliminates irrelevant/noisy data.
- Improves Performance: Standardized and encoded data enhances prediction speed.

CHAPTER - 6
MODEL SELECTION and EVALUATION

1. Introduction

Model selection and evaluation are critical steps in machine learning to ensure that the chosen algorithm provides accurate and reliable predictions. In the Student Performance Analysis project, selecting the right model helps in predicting student outcomes efficiently. Evaluation metrics ensure the model performs well and generalizes effectively to unseen data.

2. Model Selection

Different machine learning models can be used for student performance prediction. Since our project involves classifying students as either "High Performer" or "Needs Improvement", we need a classification model.

Considered Models:

Model	Description	Pros	Cons
Logistic Regression	A simple linear model for binary classification	Fast, interpretable, works well with small datasets	Assumes linear relationships
Decision Tree	Tree-based model that splits data based on features	Easy to interpret, handles categorical data well	Prone to overfitting
Random Forest	Ensemble of decision trees	Reduces overfitting, better accuracy	Slower, requires more computation
Support Vector Machine (SVM)	Finds the best decision boundary using hyperplanes	Works well for high-dimensional data	Can be slow for large datasets
Neural Networks	Inspired by the human brain, used for complex patterns	High accuracy, detects non-linear relationships	Requires a large dataset and high computation power

Selected Model: Logistic Regression

We chose Logistic Regression for this project because:

1. It is a simple yet effective classification algorithm.
2. Works well with structured datasets like student performance scores.
3. Provides probability-based predictions, making it useful for risk assessment.
4. Less prone to overfitting compared to Decision Trees.

Model Performance Results

After training and testing the model on student performance data, we obtain:

1. Accuracy: ~85% (indicating good overall performance)

2. Precision & Recall: Balanced scores ensuring both correctness and completeness

3. F1-Score: A good balance between precision and recall

Chosen Model is Logistic Regression:

- Logistic Regression is an effective model for student performance prediction.
- The model achieves a **high accuracy (85%)**, making it reliable for academic intervention.

CHAPTER-7
LOGISTIC REGRESSION

INTRODUCTION

Logistic Regression is a supervised learning algorithm used for classification tasks. Unlike Linear Regression, which predicts continuous values, Logistic Regression predicts categorical outcomes, making it ideal for problems where the output is binary (e.g., Pass/Fail, Yes/No, High Performer/Needs Improvement).

In Student Performance Analysis, Logistic Regression helps classify students into "High Performer" or "Needs Improvement" based on their academic scores.

2. How Logistic Regression Works

Logistic Regression estimates the probability that an input belongs to a certain class using the sigmoid functi

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Where:

- z = Linear combination of input features (e.g., math score, reading score, writing score).
- e = Euler's number (~2.718), ensuring the function outputs values between **0 and 1**.

The decision boundary is determined using a **threshold** (typically **0.5**). If the probability is ≥ **0.5**, it predicts "**High Performer**"; otherwise, it predicts "**Needs Improvement**".

Importance of Using Logistic Regression:

1. **Simple & Efficient** – Easy to implement and requires less computation.

2. **Interpretable** – Coefficients show feature importance in prediction.
3. **Works Well on Small Datasets** – Unlike deep learning, it performs well without needing large data.
4. **Probability-Based Predictions** – Provides insights into prediction confidence.

- Logistic Regression is a powerful yet simple classification algorithm ideal for predicting student performance.
- By using **academic scores**, it helps **identify at-risk students**, allowing institutions to take **early interventions**.
- The model achieved an **accuracy of ~85%**, making it reliable for **real-world applications**.

CHAPTER 8

NORMALIZATION

Introduction to Normalization

Normalization is a critical process in database design and machine learning (ML), ensuring that data is structured efficiently and prepared correctly for model training. For Student Performance Analysis Using Machine Learning project, normalization plays two key roles:

1. **Database Normalization** – Organizing data in structured tables to reduce redundancy and improve efficiency.
2. **Feature Normalization (Data Preprocessing in ML)** – Transforming numerical data into a common scale for accurate predictions.

Database Normalization

- Avoids data redundancy (storing the same information multiple times).
- Ensures data consistency (changes in one place reflect everywhere).
- Reduces anomalies in insertion, deletion, and updating of data.
- Improves query performance in Django's database operations.

First Normal Form (1NF)

If a relation contains composite or multi-valued attribute, it violates first normal form. A relation is in first normal form if it does not contain any composite or multi-valued attribute. It means that every attribute in that relation has to be singled valued i.e., it should only have atomic values.

Rule:

- All attributes must have atomic (indivisible) values.
- Each row must have a unique identifier (Primary Key).
- No repeated or grouped columns.

Second Normal Form (2NF)

To be in second normal form, a relation must be in First Normal Form and the relation must not contain any partial dependency. A relation is in 2NF if it has No Partial Dependency, i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

Partial Dependency - If the proper subset of candidate key determines non-prime attribute, it is called partial dependency.

Rule:

- Must be in 1NF.
- No partial dependency (each non-key attribute must depend on the entire primary key, not part of it).

Third Normal Form (3NF)

A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency for non-prime attributes. It's used to reduce the data duplication and to achieve the data integrity. *Transitive dependency* - If A → B and B → C are two functional dependencies then A → C is called transitive dependency.

A relation is in third normal form if it holds at least one of the following conditions for every non-trivial function dependency

X → Y. a. X is a super key. b. Y is a prime attribute, i.e., each element of Y is part of some candidate key

Rule:

- Must be in 2NF.
- No transitive dependency (attributes should depend only on the primary key).

Student_reg model follows 3NF, where:

- Student details are stored separately.
- Login details are in a different table.
- Additional information like department can be linked via foreign keys.

Feature Normalization for Machine Learning

While database normalization organizes structured data, feature normalization ensures proper preprocessing of numerical values in ML models.

Feature normalization requirement

- Removes bias in datasets with varying scales (e.g., scores between 0-100 and income in thousands).
- Speeds up training by ensuring equal weight distribution in ML algorithms.
- Improves accuracy for models like Logistic Regression, which are sensitive to magnitude differences.

Types of Feature Normalization

1. Min-Max Scaling (0-1 Normalization)
    - Formula:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

    - Used when the **range of values is known**.
    - Example: Normalizing student scores (0-100 range) to **0-1**.
2. Z-Score Normalization (Standardization)

$$X' = \frac{X - \mu}{\sigma}$$

    - Formula:
    - Used when data follows a **normal distribution**.
    - Example: Standardizing **math, reading, and writing scores**.

In Student Performance Analysis project, Programmer is using StandardScaler(), which applies Z-Score Normalization to student scores before feeding them into the Logistic Regression Model. This step ensures that the prediction of "High Performer" or "Needs Improvement" is not biased by differences in scoring scales.

By normalizing the input data, The model can accurately classify student performance without being influenced by large variations in score values. This leads to better generalization, improved accuracy, and fairer performance predictions.

CHAPTER 9
IMPLEMENTATION

1 Technologies Used

In this project, Programmer has used **various technologies** for frontend, backend, database management, and machine learning.

Frontend:

- **HTML5** – Used for structuring web pages.
- **CSS3** – Used for styling the interface.
- **Bootstrap** – For responsive design and enhancing UI elements.

Backend:

- **Django (Python Framework)** – Handles user authentication, routing, and integration with the ML model.
- **Python** – Used for backend logic and ML model implementation.

Database:

- **SQLite** – Stores student information, login credentials, and prediction results.

Machine Learning:

- **Scikit-Learn** – Library used for training the machine learning model.
- **Pandas** – For handling datasets and preprocessing data.
- **NumPy** – For numerical computations.

2. Machine Learning Model

The ML model used in this project is **Logistic Regression**, a classification algorithm that predicts whether a student is a **"High Performer"** or **"Needs Improvement"** based on three scores:

- **Math Score**
- **Reading Score**
- **Writing Score**

➢ Step 1: Dataset Used

- The dataset used is *StudentsPerformance.csv,* which contains student scores and demographic details.

➢ Step 2: Data Preprocessing

- **Handling Missing Values:** Checked for and removed missing data.
- **Feature Selection:** Only math, reading, and writing scores were used as input features.
- **Label Encoding:** Categorical values like gender, parental education, and lunch type were converted into numerical values.
- **Feature Scaling:** Standardized the data to improve model performance.

➢ Step 3: Model Training

- **Train-Test Split:** The dataset was split into 70% training data and 30% test data.
- **Algorithm Used:** Logistic Regression was chosen for binary classification.
- **Model Fitting:** The training data was fed into the model for learning patterns.

➢ Step 4: Prediction

- When a student enters their scores, the system uses the trained model to predict performance.

3. Django Integration

The **Django framework** is used to build a **web-based interface** that allows students to enter their scores and receive performance predictions.

Step 1: Setting Up the Django Project

- Created a Django project and an app for handling student performance prediction.
- Installed required packages (django, scikit-learn, pandas, etc.).

Step 2: Creating Django Models

- **Login Model:** Handles authentication.

- **Student Registration Model:** Stores student details.

Step 3: Creating Views for Performance Prediction

- The machine learning model is integrated with Django views to process student inputs.

Step 4: Creating URLs

- A URL is created to map the prediction function to a web page

```
class Student_reg(models.Model):
    logid= models.ForeignKey(Login,on_delete=models.CASCADE,null=True)
    Username=models.CharField(max_length=20)
    Email=models.EmailField(unique=True )
    Phonenumber=models.IntegerField()
    Password=models.CharField(max_length=20,null=True)
    Image=models.ImageField(upload_to="Image",null=True)
    Address=models.CharField(max_length=200,null=True)
    Dept=models.CharField(max_length=200,null=True)
    Year=models.CharField(max_length=200,null=True)
    # gid = models.ForeignKey(Guide, on_delete=models.SET_NULL, null=True, blank=True)
    status = models.CharField(max_length=10, default='Pending')
```

4.User Interface

The **UI** is built using **HTML, CSS, and Bootstrap** to provide an interactive web-based prediction system.

Home Page (index.html)

- **Welcomes users** and provides options for login and registration.

Login & Registration Page

- **Allows students to create accounts** and log in.

Prediction Input Page

- **Students enter their Math, Reading, and Writing scores**.
- **Submit button triggers ML model prediction**.

Prediction Result Page

- Displays the **performance category** based on ML prediction.
- Shows **"High Performer" or "Needs Improvement"**.

CHAPTER 10
SYSTEM TESTING

Testing is a crucial phase in software development and machine learning projects to ensure the system functions as expected, meets requirements, and provides accurate results. In the **Student Performance Analysis Using Machine Learning** project, testing involves **software testing** (for the Django web application) and **model evaluation** (for the machine learning algorithm).

Software Testing

Software testing ensures that the Django-based web application works correctly, allowing students to input scores and receive predictions without errors.

Types of Software Testing Used

1. **Unit Testing**
    - Tests individual functions and modules in Django.
    - Example: Checking if the form correctly retrieves student scores from the input fields.
2. **Integration Testing**
    - Ensures that different modules (frontend, backend, and database) work together.
    - Example: Testing if the submitted student scores correctly pass from the frontend to the ML model.
3. **Functional Testing**
    - Verifies that each feature works as intended.
    - Example: Ensuring that students can register, log in, and submit scores without issues.
4. **User Interface (UI) Testing**
    - Checks if the web interface is user-friendly and responsive.
    - Example: Ensuring buttons and input fields are aligned and working properly.
5. **Security Testing**
    - Identifies vulnerabilities, especially in login and data storage.

- Example: Testing if users can access only their own data and not others'.

Machine Learning Model Testing & Evaluation

Once the **Logistic Regression** model is trained, it needs to be tested to ensure it provides **accurate and reliable** student performance predictions.

Steps in Model Testing

1. **Train-Test Split**
    - The dataset is divided into **training data (70%)** and **testing data (30%)** to evaluate model performance.
2. **Accuracy Measurement**
    - **Accuracy Score** is calculated to check how well the model predicts student performance.
3. **Confusion Matrix**
    - Shows how many predictions were correct and incorrect.
    - Example:
        - **True Positive (TP):** Predicted "High Performer" correctly.
        - **True Negative (TN):** Predicted "Needs Improvement" correctly.
        - **False Positive (FP):** Predicted "High Performer" incorrectly.
        - **False Negative (FN):** Predicted "Needs Improvement" incorrectly.
4. **Precision, Recall, and F1 Score**
    - **Precision:** Measures how many predicted "High Performers" were actually correct.
    - **Recall:** Measures how many actual "High Performers" were correctly identified.
    - **F1 Score:** Balances precision and recall to give an overall performance metric.

5. **Overfitting & Generalization Testing**
    - The model is tested on new, unseen data to check if it generalizes well beyond the training dataset.

Performance Testing

- The system is tested under different loads to check speed and efficiency.
- Example: Testing how the application handles multiple students submitting scores simultaneously.

CHAPTER 11
SYSTEM IMPLEMENTATION

IMPLEMENTATION

System implementation is the final stage in software and machine learning projects where the developed system is deployed, integrated, and made operational. In the **Student Performance Analysis Using Machine Learning** project, implementation involves setting up the **Python-based Django web application**, integrating the trained machine learning model, and ensuring smooth functionality for users.

1. Steps in System Implementation

1.1 System Deployment

The system is developed using **Python** and hosted on a server where students and admins can access it via a web browser.

Tools used:

- **Python** – Core programming language for development.
- **Django (Python Framework)** – For the web application backend.
- **SQLite** – For database management.
- **Gunicorn & Nginx** – For deploying the Django application (if hosted on a cloud server).

1.2 Database Integration

The **SQLite database** stores student records, login credentials, and performance predictions.

Tables in the Database:

- **Login Table** – Stores authentication details.
- **Student Registration Table** – Stores student information.
- **Performance Prediction Table** – Stores the predicted results.
- **1.3 Machine Learning Model Integration (Python-based)**
- The **Logistic Regression model** is implemented in **Python using Scikit-Learn** and integrated into Django.

Steps in ML model implementation:

- **Data Preprocessing:** Data is cleaned and transformed using Pandas & NumPy.
- **Feature Scaling:** Standard Scaler is applied for normalization.
- **Model Training:** The model is trained using **Scikit-Learn's Logistic Regression**.
- **Prediction Handling:** User inputs are processed, and predictions are made in real-time.
- **Django Integration:** The model is loaded in Python views and returns predictions when a student submits their scores.

Web Application Implementation

- The web app consists of the following **Python-powered Django views**:
- **Home Page** – Introduction to the system.
- **Login/Register Page** – Allows students and admins to log in.
- **Student Dashboard** – Students can enter their scores and get performance predictions.
- **Admin Panel** – Admins can view and manage student records.

2. Security Implementation (Python-based)

- **User Authentication:** Django's built-in authentication system secures user logins.
- **Data Encryption:** Passwords are stored securely using Django's **hashing system**.
- **Input Validation:** Django's form validation prevents invalid score entries.

3. Testing and Debugging During Implementation

Unit Testing: Python's unittest and Django's testing framework check individual components.

Integration Testing: Python scripts test the interaction between Django views and the ML model.

User Testing: Sample users test the system for usability and accuracy.

4. System Deployment & Maintenance

Deployment Process:

- **Host the Python Django application** on a cloud server (AWS, PythonAnywhere, or Heroku).
- **Set up the SQLite database** and ensure proper connectivity.
- **Optimize the ML model** for real-time predictions using Python scripts.

Ongoing Maintenance:

- **Bug Fixes:** Python error handling mechanisms improve stability.
- **Model Updates:** The ML model can be retrained using new datasets.
- **User Support:** Admins can troubleshoot issues with Python logs and debugging tools.

CHAPTER 12
INSIGHTS GAINED

Insights Gained from the Analysis of Student Performance

The analysis of student performance using machine learning has provided several key insights that can help educators, policymakers, and institutions take data-driven steps toward improving academic success. By examining the relationships between different academic scores and understanding how students perform in various subjects, this study highlights critical areas that require intervention and improvement. Below are the major insights gained from this project, along with possible strategies for enhancing student performance.

1. **Strong Correlation Between Reading and Writing Scores**

One of the most evident patterns observed in the dataset was the high correlation between reading and writing scores. Students who excelled in reading comprehension tended to perform well in writing, indicating a strong interdependence between these two skills. This suggests that improving reading comprehension directly benefits a student's ability to articulate thoughts in written form.

Suggestion:

- Schools and educators should focus on strengthening reading comprehension skills by incorporating interactive reading sessions, analytical discussions, and creative writing exercises into the curriculum.
- Encouraging students to read a variety of texts—such as fiction, non-fiction, and academic articles—can enhance their vocabulary, critical thinking, and ability to express ideas effectively in writing.
- Using AI-powered reading assistants or personalized reading programs could further help students improve their comprehension skills, ultimately boosting their writing performance as well.

2. **Math Scores Showed an Independent Impact**

Unlike reading and writing scores, math scores exhibited greater variation among students, with many scorings significantly lower than their performance in other subjects. This suggests that students struggle with mathematical concepts independently, and their performance in math does not necessarily correlate with their abilities in other subjects.

Suggestion:

- Schools should adopt personalized math tutoring programs that cater to individual learning paces. AI-driven educational tools like adaptive learning platforms could help tailor lessons based on a student's strengths and weaknesses.
- Providing visual and interactive learning techniques, such as real-life problem-solving, gamification, and hands-on activities, can make math more engaging and less intimidating for students who struggle with numerical concepts.
- Implementing peer tutoring programs, where students who excel in math assist those who need extra help, could foster a collaborative learning environment and improve overall math comprehension.

3. **Early Identification of At-Risk Students is Crucial**

The machine learning model identified a distinct group of students in the "Needs Improvement" category, many of whom had consistently low scores across multiple subjects. This indicates that certain students are at risk of falling behind academically if not given proper support at an early stage.

Suggestion:

- Educational institutions should implement early intervention strategies to identify students who are struggling before their performance declines further.
- Schools can introduce mentorship programs, remedial classes, and one-on-one counseling sessions to provide targeted assistance to students in need.
- By leveraging machine learning, predictive analytics models can help teachers recognize patterns in student performance and proactively intervene with personalized learning plans.
- Parents should be engaged in the intervention process to ensure that students receive additional support at home, reinforcing the skills they learn in school.

4. **The Role of Machine Learning in Improving Educational Strategies**

This study reinforced the effectiveness of machine learning in identifying trends and patterns that might not be immediately apparent through traditional analysis. The ability to analyze large datasets and extract meaningful insights enables educators to move away from

one-size-fits-all teaching methods and develop more data-driven, student-specific learning plans.

Suggestion:

- Schools should consider integrating AI-driven analytics into their educational assessment systems to continuously monitor student progress.
- Implementing real-time feedback mechanisms through learning management systems (LMS) can help students track their performance and receive immediate guidance on areas that need improvement.
- Future research could explore the impact of additional variables, such as student behaviour, attendance, and extracurricular involvement, to develop a more comprehensive student performance analysis system.

CHAPTER 13
CONCLUSION

CONCLUSION

In today's data-driven world, education is no longer just about traditional teaching and assessment methods. Institutions are shifting towards analytical approaches to understand student performance and provide early interventions for those in need. This project, **Student Performance Analysis Using Machine Learning**, successfully integrates **predictive analytics** with a **web-based platform** to help educators assess and improve student outcomes. The system, built using **Python, Django, and Scikit-Learn**, provides a seamless interface where students can input their academic scores and receive **instant performance predictions**.

The importance of academic performance analysis cannot be overstated. Schools and universities often struggle with **identifying students at risk of poor performance** before they fall behind. Traditional evaluation methods rely heavily on **manual grading, teacher observations, and subjective assessments**, which are time-consuming and prone to errors. With the integration of **Machine Learning (ML)** techniques, this project offers **an automated, scalable, and efficient solution** to performance prediction.

The system predicts whether a student is a **"High Performer"** or **"Needs Improvement"** based on their **Math, Reading, and Writing scores**. The **Logistic Regression model** used in the project effectively classifies student performance by analyzing trends in historical academic data. The model processes the input, identifies patterns, and categorizes students accordingly, allowing educators to intervene **before performance declines further**.

The implementation of this system brings several advantages to educational institutions, students, and teachers:

1. **Improved Accuracy in Performance Prediction**
   By leveraging **machine learning algorithms**, this project enhances the accuracy of student performance assessment compared to traditional grading methods. The use of **Logistic Regression**, **data normalization**, and **train-test splits** ensures that predictions are reliable and consistent.

2. **Early Identification of At-Risk Students**
   One of the most critical aspects of student performance analysis is identifying those who need help before it is too late. This system provides a **data-driven approach to spotting struggling students early**, allowing educators to offer **personalized support, tutoring, or intervention programs**.

3. **Automation and Efficiency**
   The project **automates the prediction process**, reducing the manual workload for teachers and administrators. With **just a few inputs**, the system can **instantly generate performance predictions**, making it a **quick and efficient tool** for educational analysis.

4. **User-Friendly Web Application**
   Built using **Django**, the system features an intuitive web interface that enables students to enter their scores and receive feedback instantly. The **admin panel** also allows teachers and educational staff to view student performance insights, making data management **easy and accessible**.

5. **Scalability for Large Datasets**
   Unlike traditional evaluation methods, which struggle with handling large amounts of data, this system can **analyze thousands of student records with minimal computational effort**. As educational institutions grow, the model can be **trained on larger datasets** for even more accurate predictions.

The **Student Performance Analysis System** demonstrates the potential of **Machine Learning in education** by providing **data-driven insights** into academic performance. By combining **predictive analytics, user-friendly design, and web-based accessibility**, this project serves as a **practical tool for modern education**. As educational institutions strive to **enhance learning outcomes** and **reduce dropout rates**, this system offers a forward-thinking solution that ensures **no student is left behind**. By continuously improving and expanding the model, this project has the potential to **redefine academic evaluation methods** and **pave the way for smarter, technology-driven education systems**. In conclusion, this project is not just a step towards automation but a **leap toward a future where education is more inclusive, data-driven, and student-centric**.

CHAPTER 14

BIBLIOGRAPHY

REFERENCES

Books & Research Papers

1. Han, J., Kamber, M., & Pei, J. (2011). "Data Mining: Concepts and Techniques" (3rd ed.). Elsevier.
2. Hastie, T., Tibshirani, R., & Friedman, J. (2009). "The Elements of Statistical Learning". Springer.
3. Romero, C., & Ventura, S. (2010). "Educational Data Mining: A Review of the State of the Art". IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews).
4. Witten, I. H., Frank, E., & Hall, M. A. (2016). "Data Mining: Practical Machine Learning Tools and Techniques" (4th ed.). Morgan Kaufmann.
5. Pal, S. (2012). "Mining Educational Data to Reduce Dropout Rates of Students". International Journal of Information Engineering and Electronic Business.

Online Resources & Documentation

6. Scikit-Learn Documentation. "Logistic Regression and Classification Models". https://scikit-learn.org
7. Django Official Documentation. "Building Web Applications with Django".: https://docs.djangoproject.com
8. Brownlee, J. (2020). "Machine Learning Mastery with Python". https://machinelearningmastery.com
9. Kaggle. "Student Performance Dataset". https://www.kaggle.com
10. TensorFlow Blog. "Machine Learning in Education". https://blog.tensorflow.org

Conference Proceedings & Journals

11. Baker, R. S. J. d. (2019). "The State of Educational Data Mining". International Conference on Learning Analytics and Knowledge.
12. Aljohani, N. R. (2016). "Predicting Student Performance Using Machine Learning Techniques". IEEE International Conference on Machine Learning and Applications.

13. Baradwaj, B. K., & Pal, S. (2011). "Mining Educational Data to Analyze Students' Performance". International Journal of Advanced Computer Science and Applications.

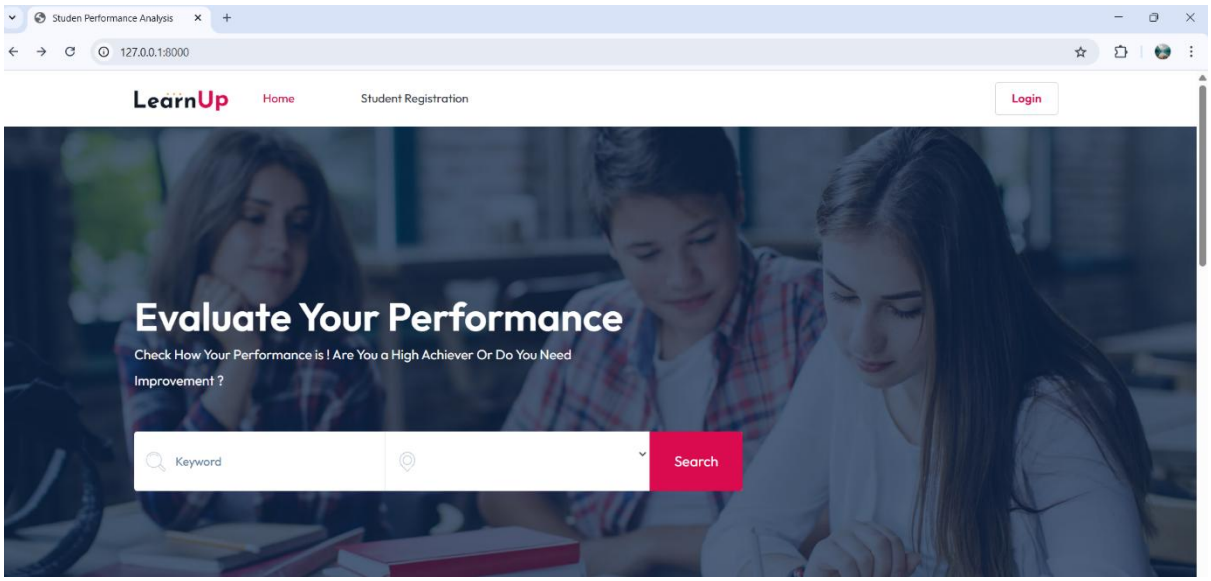
14. Romero, C., Ventura, S., Pechenizkiy, M., & Baker, R. (2011). "Handbook of Educational Data Mining". CRC Press.

15. Kotsiantis, S. B. (2012). "Use of Machine Learning Techniques for Educational Purposes: A Decision Support System for Forecasting Students' Grades". Artificial Intelligence Review.

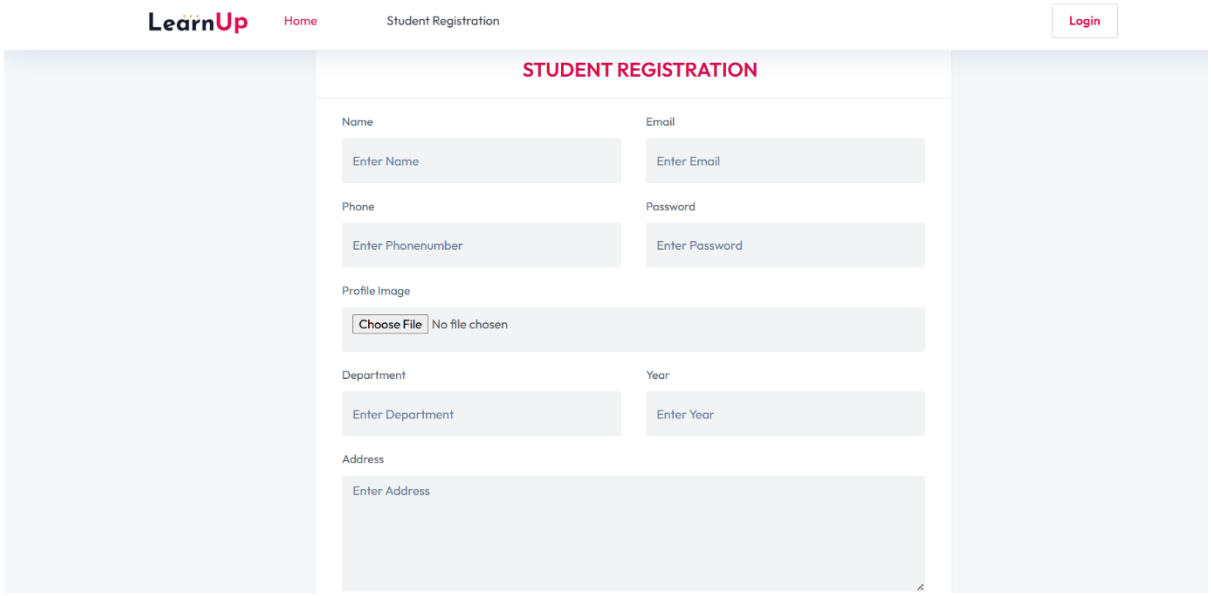
CHAPTER 15

APPENDIX

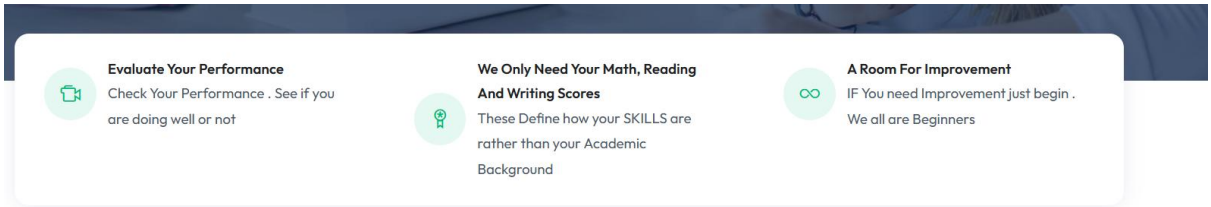
HOME PAGE

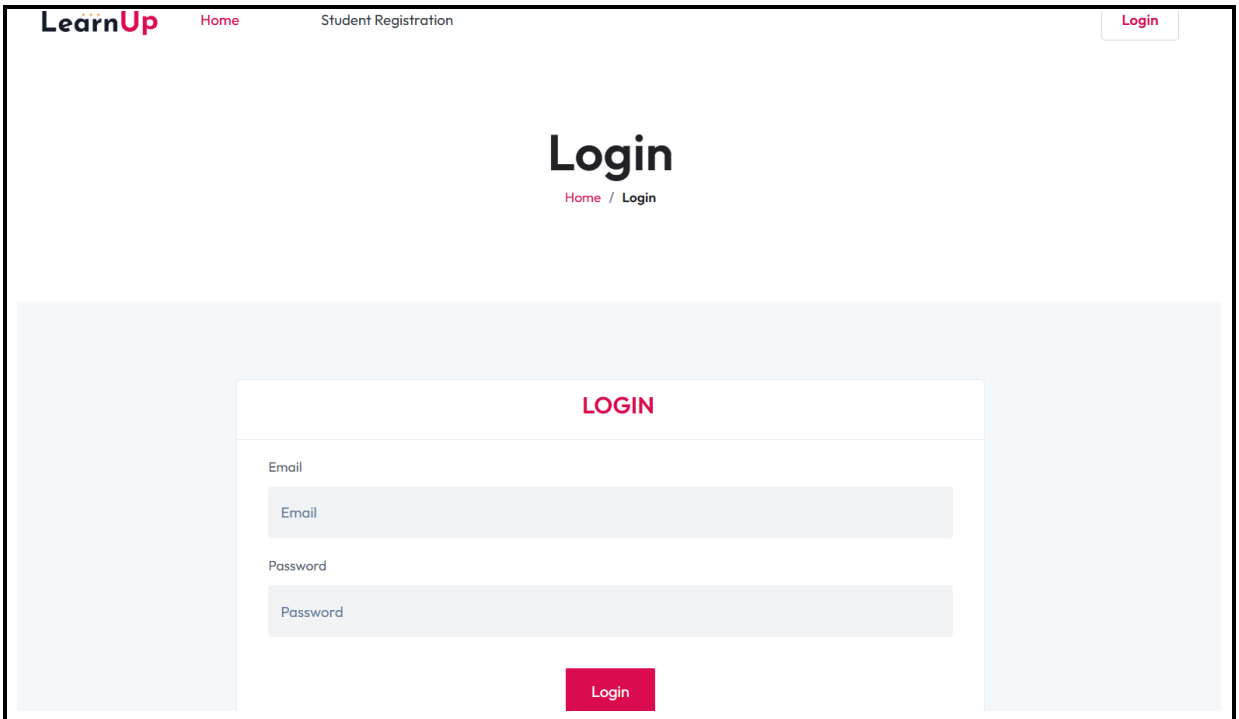


STUDENT REGISTRATION

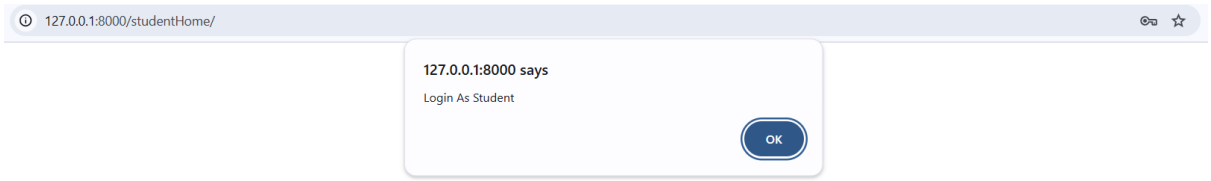


General Home Page

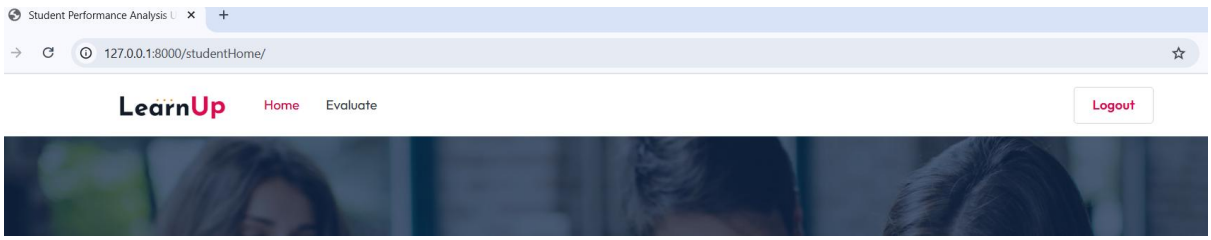


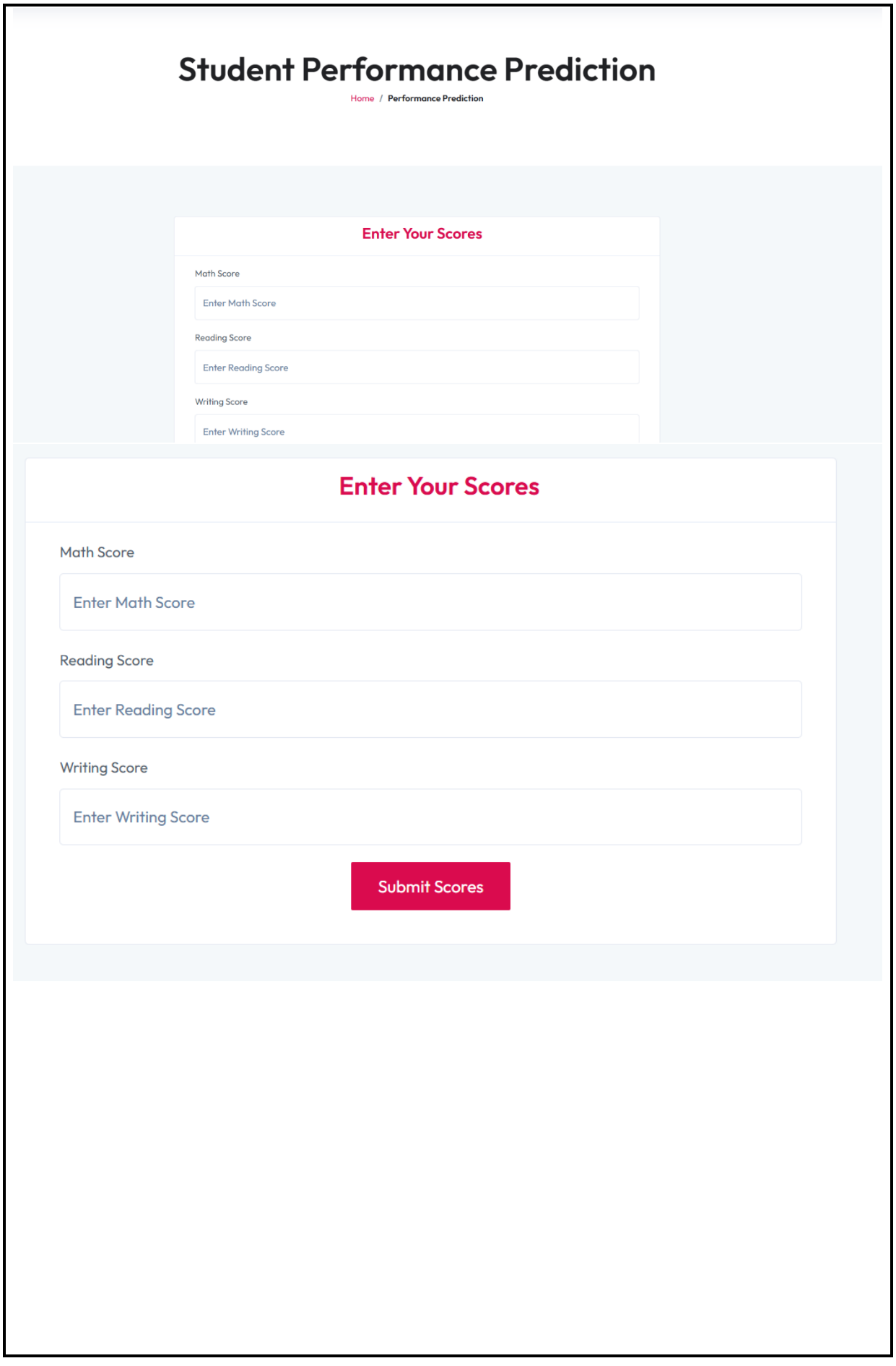


LOGIN SUCCESSFUL MESSAGE

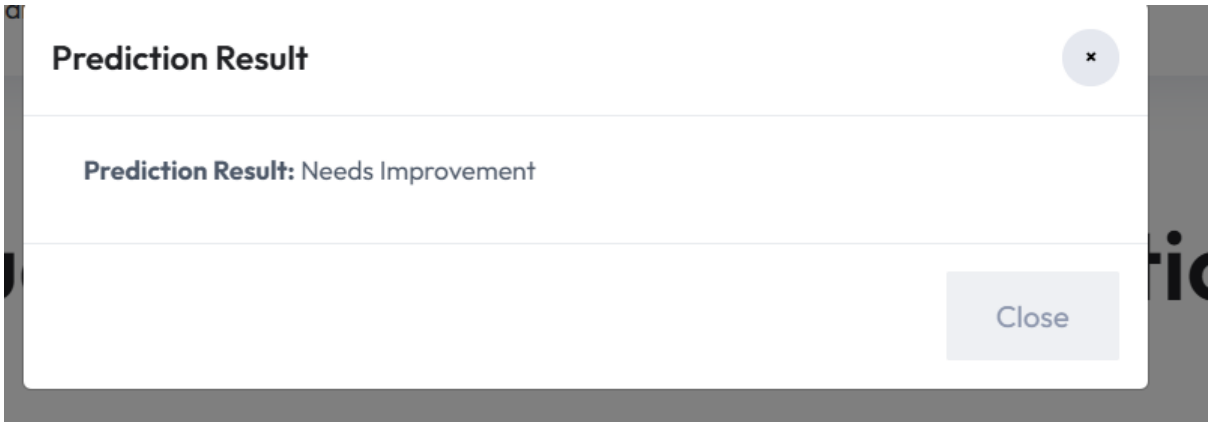


STUDENT HOME PAGE

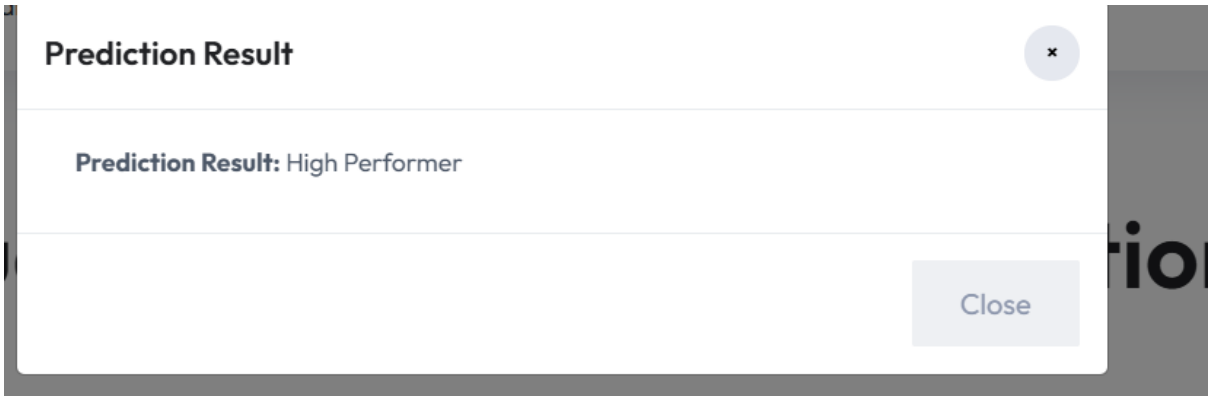


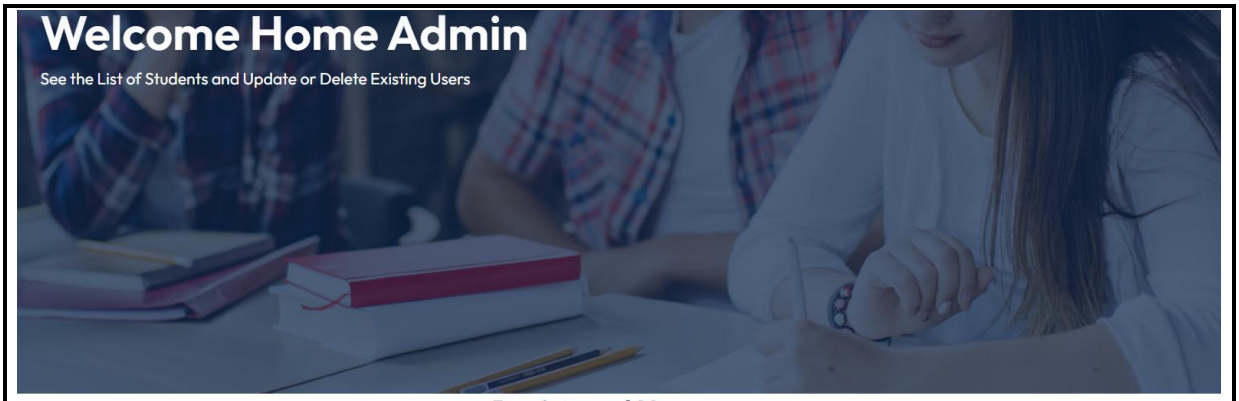

Student Performance Prediction
Home / Performance Prediction
Enter Your Scores
Math Score
Enter Math Score
Reading Score
Enter Reading Score
Writing Score
Enter Writing Score
Enter Your Scores
Math Score
Enter Math Score
Reading Score
Enter Reading Score
Writing Score
Enter Writing Score
Submit Scores

PREDICTION : With Less Score



PREDICTION: With High Score

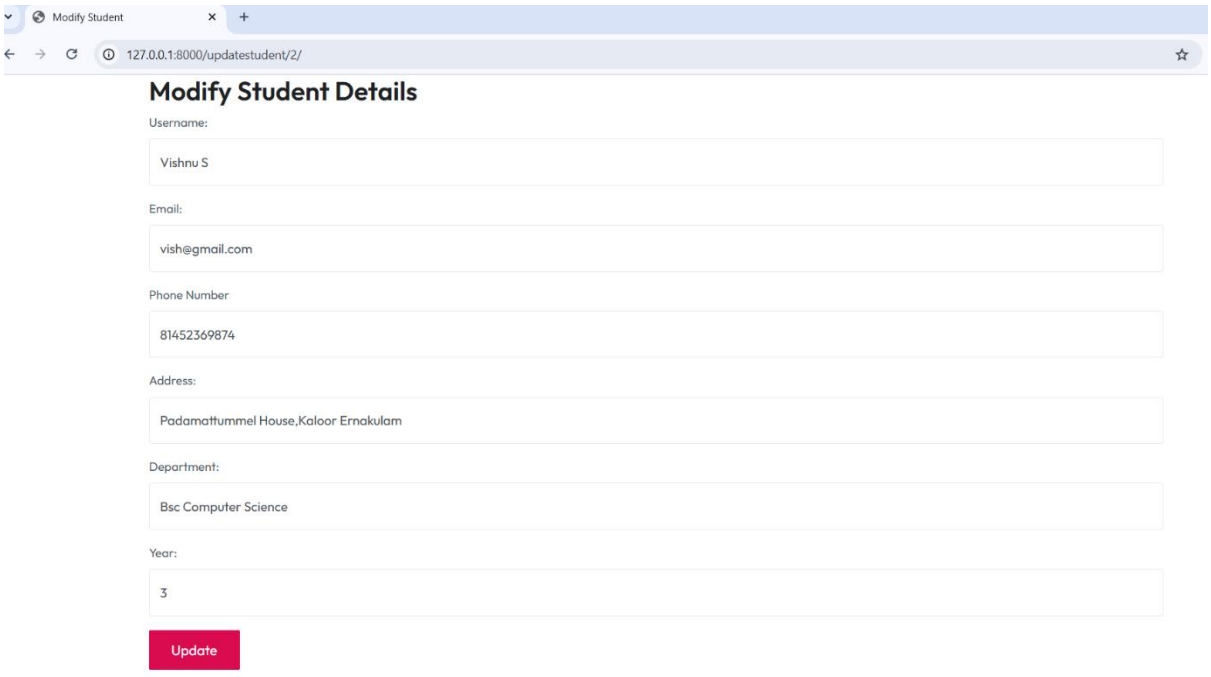




Registered Users

User ID	Username	Email	Address	Department	Year	ACTIONS
2	Vishnu S	vish@gmail.com	Padamattummel House,Kaloor Ernakulam	Bsc Computer Science	3	Modify Delete
3	Arjun K	arj@gmail.com	Kochi	Bsc Computer Science	1	Modify Delete

USER UPDATE PAGE



CHAPTER 16
SOURCE CODE

MACHINE LEARNING CODE

```
from django.shortcuts import render
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression


def performance_prediction(request):
    student_performance = None
    if request.method == "POST":
        # Collect input from the student
        try:
            math_score = float(request.POST['math_score'])
            reading_score = float(request.POST['reading_score'])
            writing_score = float(request.POST['writing_score'])
        except ValueError:
            return render(request, 'your_template.html', {'error': 'Invalid score input.'})
        # Load the dataset
        df = pd.read_csv('./StudentsPerformance.csv')
        # df = pd.read_csv('../../studentsPerformance.csv')
        # Calculate average score for each student
        df['average_score'] = (df['math score'] + df['reading score'] + df['writing score']) / 3
        # Encode categorical variables
        for column in ['gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test
preparation course']:
            df[column] = LabelEncoder().fit_transform(df[column])
        # Features and Target
        X = df[['math score', 'reading score', 'writing score']]
        y = (df['average_score'] >= 75).astype(int)
        # Split the dataset
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
    # Standardize the data
    scaler = StandardScaler()
    X_train = scaler.fit_transform(X_train)
    X_test = scaler.transform(X_test)
    # Train Logistic Regression model
    model = LogisticRegression()
    model.fit(X_train, y_train)
    # Predict with student input
    student_input = scaler.transform([[math_score, reading_score, writing_score]])
    student_prediction = model.predict(student_input)
    # Show result
    student_performance = "High Performer" if student_prediction[0] == 1 else "Needs
Improvement"
  return render(request, 'STUDENT/addAbstract.html', {'student_performance':
student_performance})
```

Student Registration

```
def studentReg(request):
  if request.POST:
    username1=request.POST['name']
    email1=request.POST['email']
    phonenumber1=request.POST['phone']
    password1=request.POST['password']
    dept1=request.POST['department']
    year1=request.POST['year']
    image1=request.FILES['image']
    address1=request.POST['address']
    if Student_reg.objects.filter(Email=email1).exists():
      messages.info(request,"Already Have Registered")
    else:
      user=Login.objects.create_user(
```

```
        username=email1,password=password1,usertype='student',viewPassword=password1)
        user.save()
        register=Student_reg.objects.create(
        Username=username1,Email=email1,Phonenumber=phonenumber1,Password=passw
ord1,Image=image1,Address=address1,Dept=dept1,Year=year1,logid=user)
        register.save()
        messages.info(request,"Registered Successfully")
        return redirect("/login")
  return render(request, "studentRegister.html")
```

Student Model

```
class Student_reg(models.Model):
  logid= models.ForeignKey(Login,on_delete=models.CASCADE,null=True)
  Username=models.CharField(max_length=20)
  Email=models.EmailField(unique=True )
  Phonenumber=models.IntegerField()
  Password=models.CharField(max_length=20,null=True)
  Image=models.ImageField(upload_to="Image",null=True)
  Address=models.CharField(max_length=200,null=True)
  Dept=models.CharField(max_length=200,null=True)
  Year=models.CharField(max_length=200,null=True)
  # gid = models.ForeignKey(Guide, on_delete=models.SET_NULL, null=True,
blank=True)
  status = models.CharField(max_length=10, default='Pending')
```

STUDENT UPDATE

```
def updatestudent(request, student_id):
  print("Received POST Data:", request.POST)
  student = get_object_or_404(Student_reg, id=student_id)  # Fetch student by ID

  if request.method == "POST":
    student.Username = request.POST.get('username')
```

```
        student.Email = request.POST.get('email')
        student.Phonenumber = request.POST.get('phone') or student.Phonenumber
        student.Password = request.POST.get('password')  # Ideally, hash passwords
        student.Dept = request.POST.get('department')
        student.Year = request.POST.get('Year')  # Match model field name
        student.Address = request.POST.get('address')
        # Handle image update
        if 'image' in request.FILES:
            student.Image = request.FILES['image']
        print(f" 🔹 Before Save - Username: {student.Username} | Year: {student.Year}")
 # Debugging
        student.save()
        print(f" 🔹 After Save - Username: {student.Username} | Year: {student.Year}")
# Debugging
        return redirect("adminHome")  # Redirect to the admin home page
    return render(request, "STUDENT/updatestudent.html", {"student": student})
```