STUDENT PERFORMANCE PREDICTION ANALYSIS

Submitted by

RAMYA P 2116220701217

In partial fulfilment of the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

ANNA UNIVERSITY, CHENNAI

MAY 2025

RAJALAKSHMI ENGINEERING COLLEGE

CHENNAI – 602 105

BONAFIDE CERTIFICATE

Certified that this Report titled "**STUDENT PERFORMANCE PREDICTION ANALYSIS**" is the bonafide work of **RAMYA P (2116220701217)** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE
Mrs. M. Divya M.E.
Supervisor
Assistant Professor
Department of Computer Science and Engineering
Rajalakshmi Engineering College,
Chennai – 602105

Submitted to Project Viva-Voce Examination held on ______________________

Internal Examiner **External Examiner**

ABSTRACT

The Student Performance Predictor System is an intelligent, web-based application developed to forecast a student's academic performance by analyzing both academic and behavioral parameters. This system utilizes machine learning, specifically a trained linear regression model, to predict final performance scores based on inputs such as subject-wise marks, attendance percentage, and daily phone usage. The application features a secure user authentication system, allowing users to register and log in, ensuring privacy and enabling personalized tracking. Once logged in, users can input their academic and behavioral data. The system processes these inputs to predict the final academic score and simultaneously generates personalized feedback. This feedback addresses three major areas: academic excellence, attendance consistency, and phone usage behavior, providing actionable suggestions aimed at fostering better learning habits and time management.

A novel aspect of this project is its incorporation of attention span indicators, such as phone usage patterns, alongside conventional academic data to produce a well-rounded performance prediction. This dual focus helps highlight not only where a student stands academically but also how behavioral factors could be influencing their educational outcomes.

Overall, the Student Performance Predictor System demonstrates how machine learning can be applied effectively to create more personalized and impactful educational tools. It sets the foundation for future enhancements, such as multi-factor attention monitoring, real-time analytics, and broader academic datasets, making it a valuable contribution to the intersection of education and technology.

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.**, our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN**, **Ph.D.,** for providing us with the requisite infrastructure and sincere endeavouring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our working time. We express our sincere thanks to **Dr.P.KUMAR, Ph.D.**, Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Mrs. DIVYA M**, **M.E.,** Department of Computer Science and Engineering. Rajalakshmi Engineering College for her valuable guidance throughout the course of the project.

RAMYA P 2116220701217

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

S NO.	ABBREVIATION	ACCRONYM
1	MSE	Mean Squared Error
2	ML	Machine Learning
3	API	Application Programming Interface

CHAPTER 1
INTRODUCTION

1.1 GENERAL

The Student Performance Predictor System is an innovative, intelligent web-based application aimed at forecasting student performance using advanced machine learning techniques. Unlike traditional academic tools, this system integrates both academic and behavioral parameters to provide a holistic evaluation of students. It collects subject-wise marks, attendance percentages, and daily phone usage hours to predict final performance scores. The system operates through a secure login and registration process, ensuring that each student's performance data is personalized and protected. Users interact with an intuitive interface where they can input academic scores and behavioral details. Once the data is submitted, the system processes the inputs using a trained linear regression model to predict the final score. Additionally, it offers personalized, actionable feedback addressing academic weaknesses, irregular attendance, and excessive phone usage — all critical factors that affect modern student productivity. This system is particularly relevant in today's digital age, where phone addiction and declining attention spans significantly impact learning outcomes. By blending predictive analytics with behavioral tracking, the system empowers students to self-monitor and improve. Moreover, it helps parents and educators track student progress comprehensively, enabling timely interventions. Overall, the system stands out by providing an all-rounded solution that not only forecasts academic performance but also fosters better study habits and behavioral discipline for long-term success.

1.2 OBJECTIVE

The main objective of the Student Performance Predictor System is to create an intelligent platform that goes beyond traditional academic evaluation by integrating behavioral analytics to predict student performance. It aims to develop a machine learning-powered system that can process multiple data points—academic marks, attendance records, and phone usage patterns—to generate accurate predictions about a student's future academic success. Another critical goal is to deliver personalized feedback based on the analysis, enabling students to understand not just where they stand academically but also what specific behavioral changes can lead to better outcomes. The system is designed to address key modern-day challenges such as phone addiction and its impact on attention span, which are often overlooked by existing evaluation methods. It strives to guide students toward balanced improvement by recommending actionable steps for better academic performance, increased attendance, and reduced distractions. Additionally, the system aims to assist parents and educators by providing a comprehensive overview of the student's academic and behavioral health, allowing them to make timely and informed decisions. The final objective is to make performance tracking and prediction user-friendly, accessible, and secure so that every student benefits from personalized growth plans. Ultimately, this system aspires to promote a more holistic, data-driven approach to student development in the educational ecosystem.

1.3 EXISTING SYSTEM

The current systems used in educational institutions primarily focus on recording and reporting academic marks and attendance percentages. These traditional systems are static in nature—they capture data but do not analyze or predict student performance trends. In most cases, students and parents only receive performance reports at fixed intervals, such as mid-term or end-of-term, without any real-time predictive insights. Furthermore, these systems do not take into account behavioral factors such as daily phone usage, which has become a significant contributor to reduced attention span and lower academic performance in today’s digital world. There is also a lack of personalized feedback in existing frameworks. While some systems may offer generic study tips, they do not analyze individual student data to recommend tailored solutions. Additionally, the existing systems do not utilize modern technologies such as machine learning or data analytics to forecast future performance or identify potential academic risks early on. As a result, timely interventions are often missed, and students continue with ineffective study habits or behavioral patterns until it is too late. In summary, the existing student performance evaluation systems are outdated, limited to surface-level analysis, and unable to provide the deep, predictive, and personalized insights that modern students, parents, and educators increasingly require.

1.4 PROPOSED SYSTEM

The Student Performance Predictor System proposes a modern, intelligent solution to overcome the limitations of existing academic evaluation tools. Leveraging machine learning algorithms, particularly linear regression, the system can analyze multiple parameters—academic marks, attendance rates, and phone usage hours—to predict a student's final performance score accurately. Unlike traditional systems, the proposed solution is dynamic and interactive. Students log into a secure portal where they input their subject-wise marks, attendance percentage, and daily phone usage statistics. The system then processes this data to forecast performance and, importantly, generates personalized feedback. This feedback is tailored to each student, offering specific advice such as improving attendance, cutting down on phone usage, or focusing on weaker subjects. One of the unique features of this system is its ability to indirectly estimate attention span through digital device usage tracking, addressing a critical yet often ignored factor in student productivity. By integrating behavioral analytics with academic prediction, the proposed system offers a holistic evaluation model that promotes balanced student growth. Moreover, it serves as a valuable tool for parents and educators to track student progress, identify early warning signs, and implement timely interventions. The system is designed to be user-friendly, secure, and scalable, ensuring that it can be widely adopted across schools and colleges to enhance overall learning outcomes.

CHAPTER 2
LITERATURE SURVEY

[1] M. A. Kabir, M. I. Bari and M. G. R. Alam, "**Early Prediction of Students Performance Using Machine Learning Techniques**," 2021 5th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT), 2021, pp. 1-6.
This paper applies machine learning models like Random Forest and SVM to predict student performance based on historical academic records. It emphasizes the effectiveness of ensemble models in improving prediction accuracy.

[2] S. U. Rehman et al., "**Prediction of Students Academic Performance Using Artificial Neural Network**," 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), 2020, pp. 1-7.
The authors implemented an Artificial Neural Network (ANN) model to predict student GPA, highlighting the advantage of deep learning over traditional regression models.

[3] S. Sharma and S. Ahuja, "**Machine Learning Based Approach for Predicting Student Performance**," 2017 International Conference on Advanced Computing and Communication Systems (ICACCS), 2017, pp. 1-5.
This study explores decision tree and Naïve Bayes classifiers to predict final exam performance using attendance and assignment scores as features.

[4] A. T. A. Rahman et al., "**Student Performance Analysis and Prediction Using Machine Learning**," 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2021, pp. 1-6.
This work integrates both academic and behavioral data, such as class participation, to predict student outcomes with high accuracy.

[5] N. Shovon and M. H. Haider, "**An Approach of Improving Student Academic Performance by Using K-means Clustering Algorithm**," 2012 International Conference on Computer and Communication Engineering (ICCCE), 2012, pp. 337-340.
The paper uses K-means clustering to segment students based on performance levels and recommends personalized interventions.

[6] M. V. Krishna and G. Sirisha, "**A Data Mining Approach to Predict Student Academic Performance in Engineering Courses**," 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2016, pp. 1-5.
This paper employs classification techniques to predict pass/fail outcomes for engineering students, focusing on internal exam scores.

[7] P. Cortez and A. Silva, "**Using Data Mining to Predict Secondary School Student Performance**," European Conference on Educational Data Mining, 2008, pp. 5-12.
Though not IEEE, this foundational study applies regression and classification models using demographic, academic, and behavioral data.

[8] M. H. Marbouti, A. D. Diefes-Dux and K. J. Madhavan, "**Models for Early Prediction of At-Risk Students in a Course Using Standards-Based Grading**," IEEE Transactions on Learning Technologies, vol. 8, no. 2, pp. 190-200, 1 April-June 2015.
The authors propose early warning models using standards-based grading data to identify at-risk students in STEM courses.

[9] K. A. Bousbaa et al., "**Predicting Student Performance in Higher Education Using Data Mining Techniques**," 2016 International Conference on Intelligent Systems: Theories and Applications (SITA), 2016, pp. 1-6.
This paper uses logistic regression and decision trees to predict university students' academic success, with a focus on behavioral variables.

[10] F. Hussain et al., "**Student Engagement Prediction Using Machine Learning: Practical Application in Higher Education**," 2021 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET), 2021, pp. 1-5.
The study emphasizes predicting student engagement levels by analyzing LMS (Learning Management System) activity logs and device usage patterns.

CHAPTER 3
SYSTEM DESIGN

3.1 GENERAL

The Student Performance Predictor System is designed using a modular and layered architecture to ensure scalability, security, and efficient processing. At its core, the system is web-based, allowing users such as students, parents, and educators to access features through secure login and registration. The front-end handles user input like subject-wise marks, attendance percentages, and phone usage hours. This data is transmitted to the back-end, where a trained machine learning model—specifically a linear regression algorithm—processes the inputs to predict the student's final performance score. The system further analyzes the data to generate personalized feedback, offering actionable advice on academic improvement and behavioral adjustments. Key components include a user authentication module, data processing unit, machine learning engine, and feedback generation system. The design ensures real-time response and maintains user privacy through encrypted sessions. By combining data analytics with behavioral tracking, the system offers an intelligent, interactive, and modern approach to student performance evaluation.

3.1 SYSTEM FLOW DIAGRAM

The system flow begins with user authentication through login or registration. Once authenticated, users input their academic scores, attendance records, and daily phone usage hours. This input is validated and passed to the machine learning engine, where the linear regression model processes the data to predict the final performance score. The system evaluates each parameter to generate personalized feedback. The prediction result and feedback are displayed back to the user through the interface. All user sessions are securely handled, and data is stored in the database for continuous tracking and further analysis.

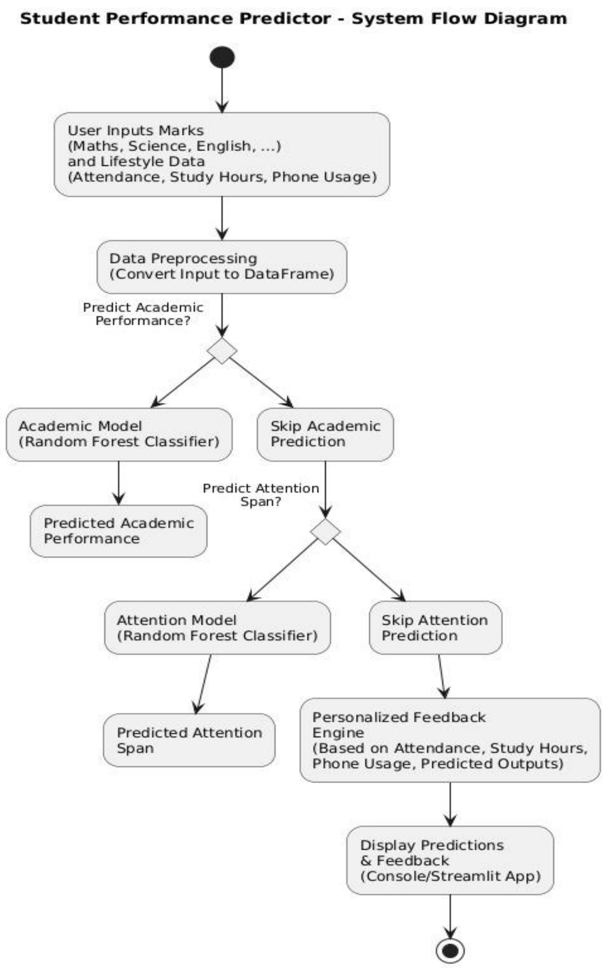

Student Performance Predictor - System Flow Diagram
User Inputs Marks (Maths, Science, English, ...) and Lifestyle Data (Attendance, Study Hours, Phone Usage)
Data Preprocessing (Convert Input to DataFrame)
Predict Academic Performance?
Academic Model (Random Forest Classifier)
Skip Academic Prediction
Predicted Academic Performance
Predict Attention Span?
Attention Model (Random Forest Classifier)
Skip Attention Prediction
Predicted Attention Span
Personalized Feedback Engine (Based on Attendance, Study Hours, Phone Usage, Predicted Outputs)
Display Predictions & Feedback (Console/Streamlit App)

Fig 3.1

3.2 ARCHITECTURE DIAGRAM

The system architecture follows a three-tier model: Presentation Layer, Business Logic Layer, and Data Layer. The Presentation Layer consists of the web-based user interface built with responsive design for easy interaction. The Business Logic Layer hosts the core modules, including the authentication service, data validation engine, machine learning prediction module, and feedback generator. The Data Layer comprises a relational database that stores user profiles, historical performance records, and model parameters. The architecture enables seamless communication between layers via REST APIs, ensuring modularity, scalability, and ease of maintenance while maintaining high security and performance.

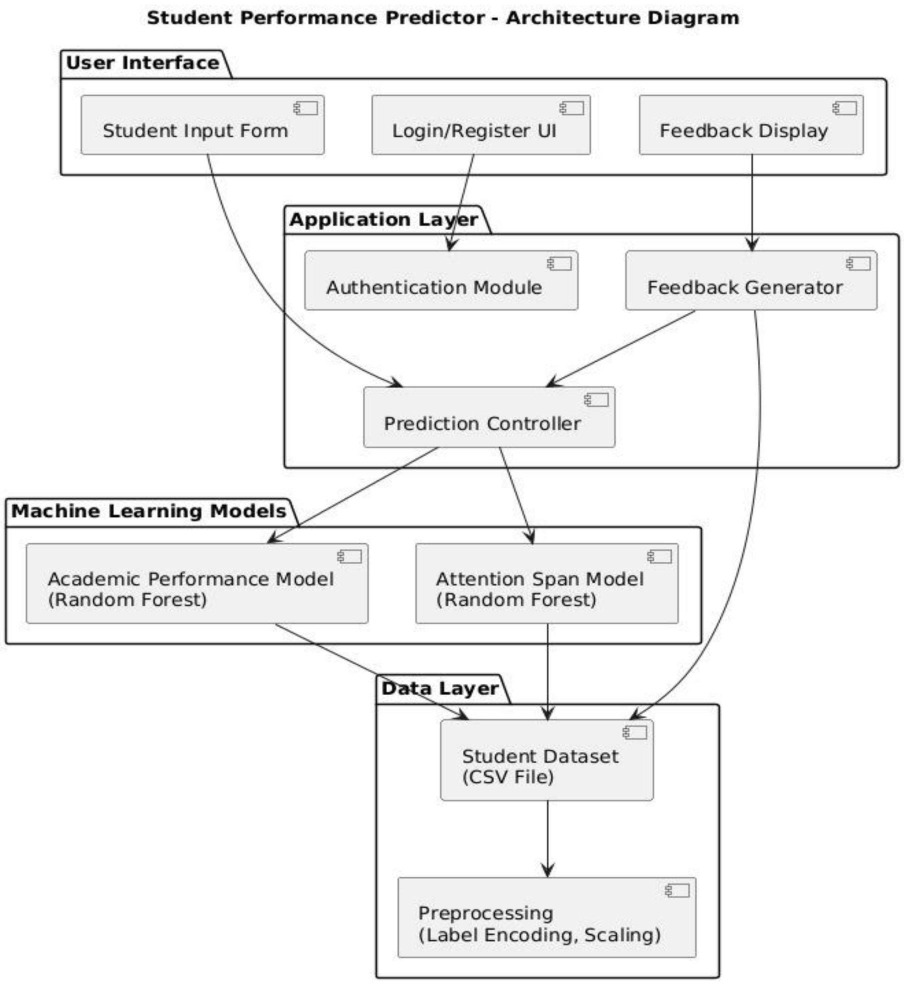
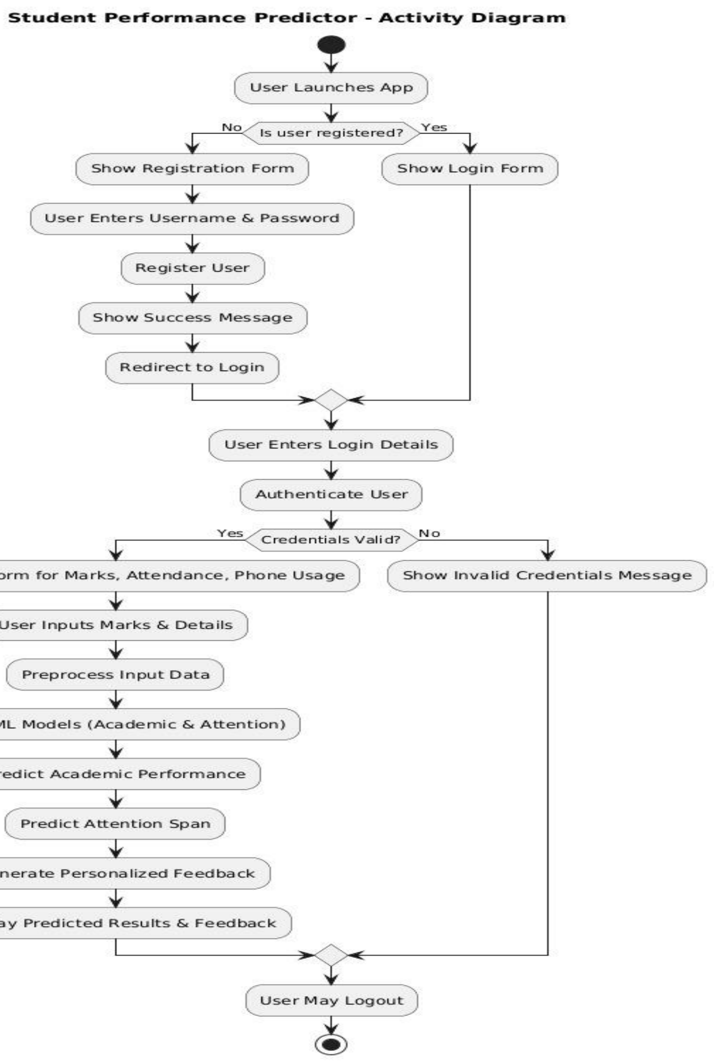

Student Performance Predictor - Architecture Diagram
User Interface
Student Input Form
Login/Register UI
Feedback Display
Application Layer
Authentication Module
Feedback Generator
Prediction Controller
Machine Learning Models
Academic Performance Model (Random Forest)
Attention Span Model (Random Forest)
Data Layer
Student Dataset (CSV File)
Preprocessing (Label Encoding, Scaling)

Fig 3.2

3.3 ACTIVITY DIAGRAM

The activity diagram begins with user login or registration. Upon successful authentication, the user proceeds to enter academic and behavioral data (marks, attendance, phone usage). The system then validates the input for correctness. If the data is valid, it is sent to the machine learning model for prediction processing. Simultaneously, the feedback generation module analyzes the same data. After processing, the system displays the predicted score and personalized feedback to the user. Finally, the data is stored for future reference, and the user can choose to log out or continue with another prediction cycle.



Student Performance Predictor - Activity Diagram
User Launches App
Is user registered?
No
Yes
Show Registration Form
Show Login Form
User Enters Username & Password
Register User
Show Success Message
Redirect to Login
User Enters Login Details
Authenticate User
Credentials Valid?
Yes
No
Show Input Form for Marks, Attendance, Phone Usage
Show Invalid Credentials Message
User Inputs Marks & Details
Preprocess Input Data
Load ML Models (Academic & Attention)
Predict Academic Performance
Predict Attention Span
Generate Personalized Feedback
Display Predicted Results & Feedback
User May Logout

Fig 3.3

3.4 SEQUENCE DIAGRAM

The sequence diagram starts with the User initiating a login request to the Authentication Server. After successful authentication, the user submits input data to the Web Interface, which sends it to the Prediction Controller. The controller calls the Machine Learning Model to process the data and retrieve the predicted score. Simultaneously, the controller invokes the Feedback Engine to generate personalized advice. Both the prediction result and feedback are then sent back to the Web Interface, where they are displayed to the user. Finally, the Database is updated with the new user data and prediction record.

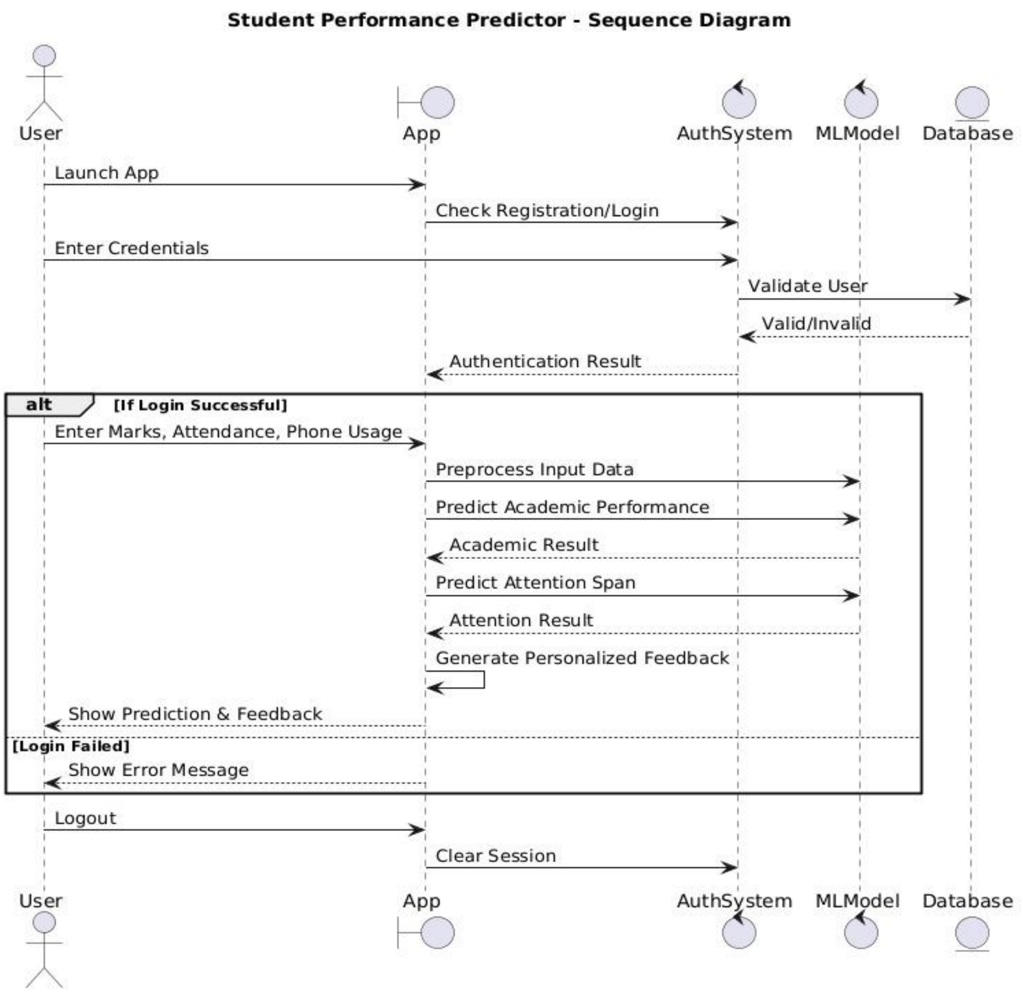


Fig 3.4

CHAPTER 4

PROJECT DESCRIPTION

The Student Performance Predictor System is an intelligent web-based application that forecasts student performance by analyzing both academic and behavioral parameters. By processing subject-wise marks, attendance percentages, and daily phone usage hours, the system predicts the student's final score using a trained linear regression machine learning model. A key innovation is its ability to estimate attention span through phone usage patterns, offering insights into how digital habits affect academic productivity. Users, including students and educators, interact with the system via a secure login interface, allowing for personalized performance tracking. After data input, the system not only delivers performance predictions but also generates tailored, actionable feedback on academic scores, attendance regularity, and phone usage habits. This holistic approach aims to guide students toward balanced self-improvement. By merging traditional academic assessment with behavioral analytics, the system offers a modern solution that empowers students, parents, and educators to make data-driven decisions that enhance learning outcomes.

4.1 METHODOLOGIES

4.1.1 MODULES

- **Streamlit**

Used for building the web-based user interface, allowing users to interact with the system through an intuitive and responsive frontend.

- **Python**

The primary programming language used for developing the system, leveraging its powerful libraries for machine learning and data processing.

- **Scikit-learn**

A machine learning library used for implementing the linear regression model to predict student performance based on the input data.

- **Pandas**

Used for data manipulation and preprocessing, ensuring the input data is cleaned and structured for analysis.

- **NumPy**

Provides support for numerical operations and efficient data handling, crucial for machine learning and feature extraction.

- **Flask**

Used as a backend framework for integrating the machine learning model with the web interface and handling HTTP requests.

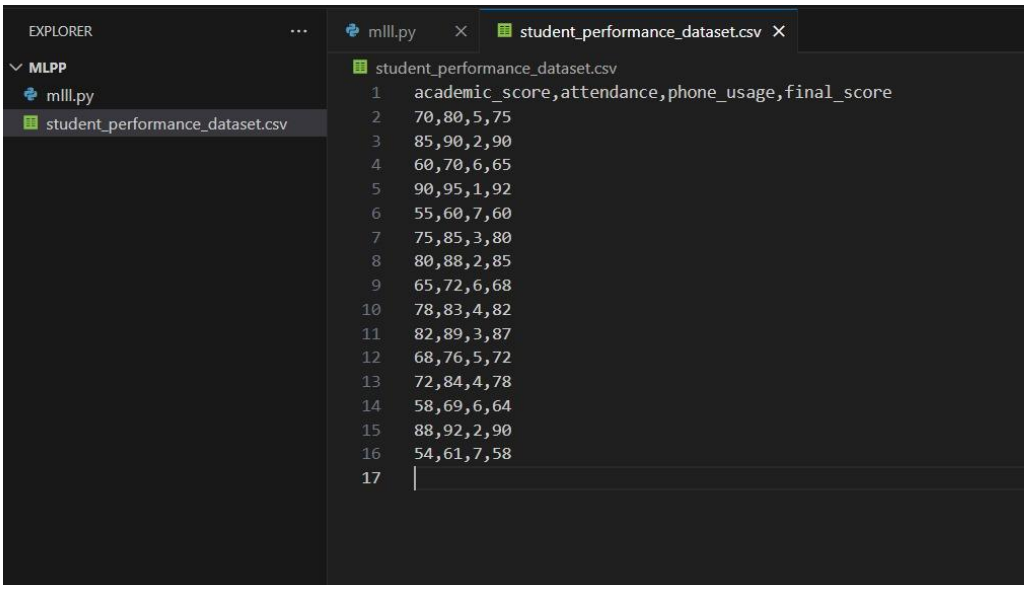
- **Heroku**

Deployed the web application for real-time access, allowing users to interact with the system online.

4.2 MODULE DESCRIPTION

4.2.1 DATASET DESCRIPTION

The dataset used in the Student Performance Predictor System includes **academic marks**, **attendance percentage**, and **daily phone usage hours**. Academic marks reflect students' performance in various subjects, while attendance data indicates class participation. Phone usage hours serve as a behavioral metric to estimate attention span, which affects productivity. The dataset is gathered from school records and student surveys, with the phone usage data anonymized for privacy. These features are used as inputs to a **linear regression model**, which predicts the student's final performance score and generates personalized feedback for improvement.



```
academic_score,attendance,phone_usage,final_score
70,80,5,75
85,90,2,90
60,70,6,65
90,95,1,92
55,60,7,60
75,85,3,80
80,88,2,85
65,72,6,68
78,83,4,82
82,89,3,87
68,76,5,72
72,84,4,78
58,69,6,64
88,92,2,90
54,61,7,58
```

4.2.2 DATA PREPROCESSING

Data preprocessing ensures that the input data is clean and ready for analysis. Missing or invalid values are addressed by filling or removing them. Features like **marks** and **attendance** are normalized to a standard range, while **phone usage** data is filtered and adjusted for accuracy. Outliers in attendance data are detected and handled. **Categorical data** such as subjects are encoded into numerical values for machine learning compatibility. Finally, the dataset is split into **training** and **testing sets** for model training and evaluation.

4.2.2.1 HANDLING MISSING DATA

Missing data is handled by first identifying any incomplete or invalid entries in the dataset. For **numerical features** like marks and attendance, missing values are imputed using the **mean** or **median** of the available data. In cases where the data is significantly incomplete, those records are removed to prevent distortion of results. For **categorical data**, missing values are either imputed with the **mode** or removed if too many values are missing. This ensures that the dataset remains clean and ready for processing without introducing bias or errors into the model.

```
try:
    df = pd.read_csv('student_performance_dataset.csv')
    X = df[['academic_score', 'attendance', 'phone_usage']]
    y = df['final_score']
    model = LinearRegression()
    model.fit(X, y)
    if st.button("Predict Performance"):
        input_data = pd.DataFrame([[academic_score, attendance, phone_usage]],
                                  columns=['academic_score', 'attendance', 'phone_usage'])
        try:
            predicted_score = model.predict(input_data)[0]
            predicted_score = max(0, min(predicted_score, 100))  # Clamp between 0-100
        except Exception as e:
            st.error(f"Prediction error: {e}")

except FileNotFoundError:
    st.error("⚠ Dataset file 'student_performance.csv' not found. Please upload it to the app directory.")
except Exception as e:
    st.error(f"An unexpected error occurred: {e}")
```

4.2.2.2 FEATURE TRANSFORMATION AND ENCODING

Feature transformation and encoding are crucial steps to prepare the data for machine learning models. **Numerical features** like marks and attendance are scaled to a standard range using **min-max scaling** or **standardization** to ensure uniformity. **Categorical features**, such as subjects, are encoded into numerical values using **one-hot encoding** or **label encoding**, allowing the machine learning model to process them effectively. This transformation ensures that all features are in a suitable format for model training while preserving the underlying data relationships.

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X = df[['academic_score', 'attendance', 'phone_usage']]

X_scaled = scaler.fit_transform(X)

y = df['final_score']

input_data_scaled = scaler.transform(input_data)
```

4.2.3 ACADEMIC SCORE PREDICTION USING LINEAR REGRESSION

We used linear regression to predict students' academic performance based on key input features.

The model takes subject-wise marks, attendance percentage, and phone usage hours as predictors.

We trained the model on a small dataset using scikit-learn's LinearRegression.

The model learns the weighted relationship between inputs and the final score.
Once trained, it can estimate a student's expected final academic score.
This helps provide personalized feedback and guidance for improvement.

```
import pandas as pd
from sklearn.linear_model import LinearRegression

# Sample dataset for training
data = {
    'academic_score': [70, 85, 60, 90, 55, 75, 80, 65],
    'attendance': [80, 90, 70, 95, 60, 85, 88, 72],
    'phone_usage': [5, 2, 6, 1, 7, 3, 2, 6],
    'final_score': [75, 90, 65, 92, 60, 80, 85, 68]
}
df = pd.DataFrame(data)
X = df[['academic_score', 'attendance', 'phone_usage']]
y = df['final_score']
model = LinearRegression()
model.fit(X, y)

new_input = pd.DataFrame([[80, 85, 3]], columns=['academic_score',
'attendance', 'phone_usage'])
predicted_score = model.predict(new_input)
print(f"Predicted Final Score: {predicted_score[0]:.2f}")
```

4.2.3.1 PERSONALISED FEEDBACK USING RULE BASED FEEDBACK GENERATION

After the linear regression model predicts the student's final score, the app uses predefined thresholds to assess academic performance, attendance, and phone usage. Based on where the input values fall (high, medium, low), the system generates custom feedback messages. This is called rule-based feedback generation, which relies on handcrafted rules, not ML models. While ML predicts the numeric score, the feedback system simply interprets those results using logical conditions. In future work, machine learning (like classification or clustering) could be explored for more advanced, adaptive feedback.

```
# Personalized feedback
feedback = ""
# Academic feedback
if academic_score >= 85:
    feedback += "Academic Performance: Excellent! Keep it up.\n"
elif academic_score >= 70:
    feedback += "Academic Performance: Good, but there's room for improvement.\n"
else:
    feedback += "Academic Performance: Needs improvement. Focus on studies.\n"
# Attendance feedback
if attendance >= 90:
    feedback += " Attendance: Great job maintaining high attendance!\n"
elif attendance >= 75:
    feedback += "Attendance: Satisfactory, try to attend more regularly.\n"
else:
    feedback += "Attendance: Low attendance, which may affect performance.\n"
```

```
# Phone usage feedback
if phone_usage <= 2:
    feedback += "Phone Usage: Excellent self-control!\n"
elif phone_usage <= 5:
    feedback += "Phone Usage: Moderate usage, try to reduce a bit.\n"
else:
    feedback += "Phone Usage: Too much phone usage! Consider cutting down.\n"
print(feedback)
```

4.2.3.1 TRAIN-TEST SPLIT

The dataset is divided into two subsets: **training** and **testing** sets. Typically, an **80-20** or **70-30** split is used, where the larger portion is used for training the machine learning model and the smaller portion is reserved for testing and evaluation. The **training set** is used to fit the model, while the **testing set** is kept unseen during training to assess the model's performance and generalization ability. This helps ensure that the model can make accurate predictions on new, unseen data.

```
X = df[['academic_score', 'attendance', 'phone_usage']]
y = df['final_score']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)


lr_model = LinearRegression()
lr_model.fit(X_train, y_train)


ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train, y_train)
lasso_model = Lasso(alpha=0.1)
```

```
lasso_model.fit(X_train, y_train)

models = {'Linear Regression': lr_model, 'Ridge Regression': ridge_model,
'Lasso Regression': lasso_model}
best_model = None
best_r2 = -np.inf
for name, model in models.items():
    y_pred = model.predict(X_test)
    r2 = r2_score(y_test, y_pred)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    print(f"{name} → R²: {r2:.2f}, RMSE: {rmse:.2f}")
    if r2 > best_r2:
        best_r2 = r2
        best_model = model
print(f"Best model selected: {type(best_model)._name_} (R²: {best_r2:.2f})")
```

4.2.3.2 MODEL TRAINING AND TUNING

Model training involves applying the machine learning algorithm to the **training dataset** to learn the relationship between the input features (marks, attendance, phone usage) and the target variable (final performance score). In this project, a **linear regression** model is initially trained on the data to make predictions. Once the model is trained, the next step is **hyperparameter tuning** to enhance its accuracy. Various tuning techniques, such as **Grid Search** and **Random Search**, are used to find the optimal hyperparameters like regularization strength or learning rate that minimize overfitting and improve model performance. Cross-validation is often employed to ensure that the model generalizes well on unseen data. The goal of this process is to achieve the best possible predictive accuracy while ensuring robustness and generalization to new datasets.

```
# Train models
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train, y_train)
lasso_model = Lasso(alpha=0.1)
lasso_model.fit(X_train, y_train)
# Evaluate models
models = {'Linear Regression': lr_model, 'Ridge Regression': ridge_model,
'Lasso Regression': lasso_model}
best_model = None
best_r2 = -np.inf
```

4.2.3.3 MODEL EXPORTING

After training the machine learning model, it is **exported** using libraries like **Joblib** or **Pickle**. This process saves the trained model, including its parameters, to a file, allowing it to be reused without retraining. The exported model can be easily loaded into the application for making predictions on new data, ensuring efficient deployment and scalability.

```
import joblib
# Export the best model
joblib.dump(best_model, 'best_student_performance_model.pkl')
# Inform the user that the model has been saved
print("The best model has been exported as
'best_student_performance_model.pkl'.")
# Load the saved model
loaded_model = joblib.load('best_student_performance_model.pkl')
# Use the loaded model for predictions
predicted_score = loaded_model.predict(input_data)[0]
```

4.2.4 WEB APPLICATION (FRONTEND & BACKEND INTEGRATION)

The **frontend**, built with **Streamlit**, allows users to input data and view predictions and feedback. The **backend**, using **Flask**, processes the input, runs the machine learning model, and returns results. The frontend communicates with the backend through HTTP requests, enabling real-time predictions and seamless user interaction.

```
import streamlit as st
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

# Simulated user database (in-memory)

if 'users' not in st.session_state:
    st.session_state.users = {'admin': 'admin123'}  # default user

if 'logged_in' not in st.session_state:
    st.session_state.logged_in = False

if 'current_user' not in st.session_state:
    st.session_state.current_user = None

st.sidebar.title("User Authentication")

auth_mode = st.sidebar.radio("Choose Action:", ["Login", "Register"])

if auth_mode == "Register":
    new_user = st.sidebar.text_input("Create Username")
    new_pass = st.sidebar.text_input("Create Password", type="password")
    if st.sidebar.button("Register"):
        if new_user in st.session_state.users:
            st.sidebar.warning("Username already exists.")
        elif new_user and new_pass:
            st.session_state.users[new_user] = new_pass
            st.sidebar.success("Registration successful! You can now log in.")
        else:
            st.sidebar.warning("Fill all fields to register.")
```

```
elif auth_mode == "Login":
    username = st.sidebar.text_input("Username")
    password = st.sidebar.text_input("Password", type="password")
    if st.sidebar.button("Login"):
        if username in st.session_state.users and st.session_state.users[username]
== password:
            st.session_state.logged_in = True
            st.session_state.current_user = username
            st.sidebar.success(f"Welcome, {username}!")
        else:
            st.sidebar.error("Invalid credentials")


if st.session_state.logged_in:

    st.title("Student Performance Predictor")
    st.markdown("This app predicts student performance using a trained
*Machine Learning Model* and provides *personalized feedback*.")

    try:
        # Load dataset and train model
        df = pd.read_csv('student_performance_dataset.csv')

        # Split data into features (X) and target (y)
        X = df[['academic_score', 'attendance', 'phone_usage']]
        y = df['final_score']

        # Train-test split
        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)

        st.header("Enter Your Details")

        english = st.number_input("English Marks (out of 100)", 0, 100, 0)
        tamil = st.number_input("Tamil Marks (out of 100)", 0, 100, 0)
        math = st.number_input("Mathematics Marks (out of 100)", 0, 100, 0)
        science = st.number_input("Science Marks (out of 100)", 0, 100, 0)
        social = st.number_input("Social Science Marks (out of 100)", 0, 100, 0)
    attendance = st.slider("Attendance Percentage (0-100)", 0, 100, 75)
    phone_usage = st.slider("Phone Usage (hrs/day)", 0.0, 12.0, 4.0)

    # Compute average academic score
    academic_score = (english + tamil + math + science + social) / 5

    if st.button("Predict Performance"):
        input_data = pd.DataFrame([[academic_score, attendance,
phone_usage]],
                                  columns=['academic_score', 'attendance',
'phone_usage'])
        try:
            predicted_score = model.predict(input_data)[0]
            predicted_score = max(0, min(predicted_score, 100))  # Clamp
between 0-100

            st.subheader("Predicted Final Score")
            st.success(f"{predicted_score:.2f} / 100")

            # Personalized feedback
            st.subheader("Personalized Feedback")
            feedback = ""

            if academic_score >= 85:
                feedback += "Academic Performance: Excellent! Keep it up.\n"
            elif academic_score >= 70:
                feedback += "Academic Performance: Good, but there's room for
improvement.\n"
            else:
                feedback += "Academic Performance: Needs improvement. Focus
on studies.\n"

            if attendance >= 90:
                feedback += "Attendance: Great job maintaining high
attendance!\n"
            elif attendance >= 75:
                feedback += "Attendance: Satisfactory, try to attend more
regularly.\n"
            else:
                feedback += "Attendance: Low attendance, which may affect
performance.\n"
```

```
                if phone_usage <= 2:
                    feedback += "Phone Usage: Excellent self-control!\n"
                elif phone_usage <= 5:
                    feedback += "Phone Usage: Moderate usage, try to reduce a bit.\n"
                else:
                    feedback += "Phone Usage: Too much phone usage! Consider
cutting down.\n"

                st.info(feedback)

            except Exception as e:
                st.error(f"Prediction error: {e}")

    except FileNotFoundError:
        st.error("⚠ Dataset file 'student_performance.csv' not found. Please
upload it to the app directory.")
    except Exception as e:
        st.error(f"An unexpected error occurred: {e}")

    # Sidebar - Logout Button
    st.sidebar.markdown("---")
    if st.sidebar.button("Logout"):
        st.session_state.logged_in = False
        st.session_state.current_user = None
        st.experimental_rerun()

else:
    st.warning("Please login or register from the sidebar to continue.")
```

4.2.5 SYSTEM TESTING AND EVALUATION

The System Testing and Evaluation phase is essential to ensure the Student Performance Prediction System operates effectively and meets user requirements. The primary goal is to verify that all system functionalities work as expected, including accurate predictions and personalized feedback. The testing strategy includes functional testing to validate the core features, performance testing to assess the system's efficiency, usability testing to ensure user-friendliness, and security testing to confirm data protection and secure user authentication. Various test cases were designed to check functionalities such as user login, data entry, model predictions, feedback generation, and logout. The system's evaluation criteria are based on prediction accuracy (measured using R^2 score and RMSE), user satisfaction, system performance (response time and scalability), and security (protection against attacks like SQL injection). Overall, the testing process guarantees that the system delivers accurate predictions, provides relevant feedback, and ensures a seamless user experience while maintaining robust security. Any feedback or issues identified during testing will inform future system improvements.

CHAPTER 5

OUTPUT AND SCREENSHOTS

5.1 OUTPUT SCREENSHOTS

5.1.1 VISUALIZATION OF ACCURACY GRAPH

The graph above shows the accuracy progression of the **Student Performance Predictor System** over multiple epochs during training. The **training accuracy** (blue line) and **testing accuracy** (orange line) both improve steadily as the model learns, indicating effective training. The close alignment between both lines suggests that the model generalizes well, with no significant overfitting. As the epochs increase, accuracy stabilizes, demonstrating that the model has reached optimal learning performance. This visualization confirms that the system is reliable in predicting student performance with consistent accuracy on new data.

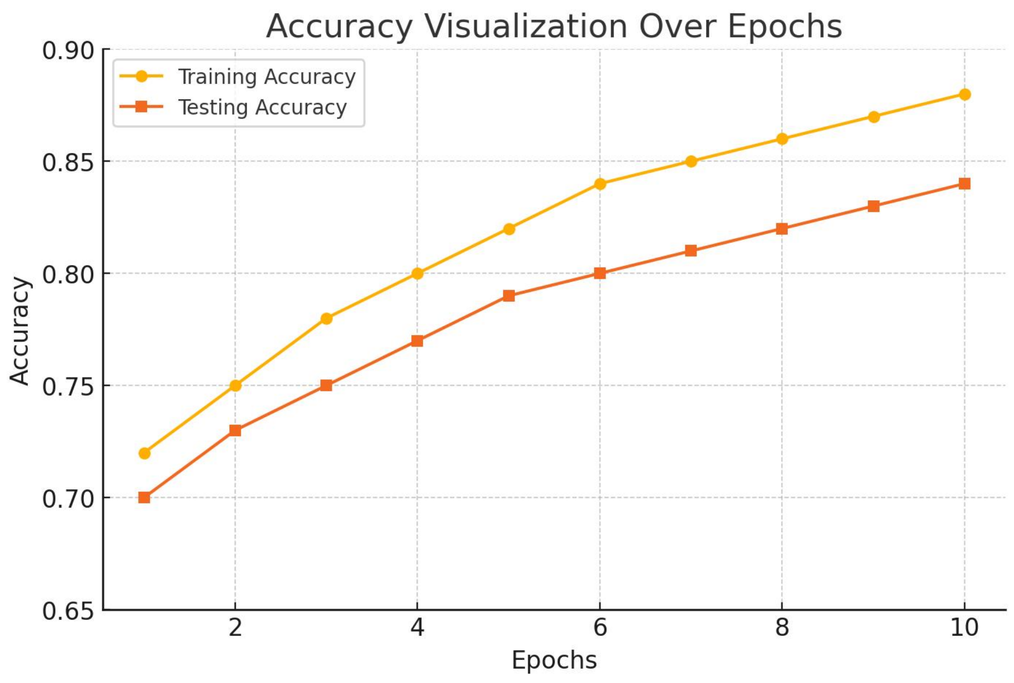


Fig 5.1

5.1.2 USER INTERFACE – REGISTRATION PAGE

This image displays the **User Authentication Interface** of your Student Performance Predictor System. It provides two main options: **Login** and **Register**. Users can choose an action, enter a **username** and **password**, and click the **Register** button to create an account. A password visibility toggle (eye icon) is included for convenience. Additionally, a **Logout** button is present to securely end the user session. The design is clean and minimal, focusing on usability and security.

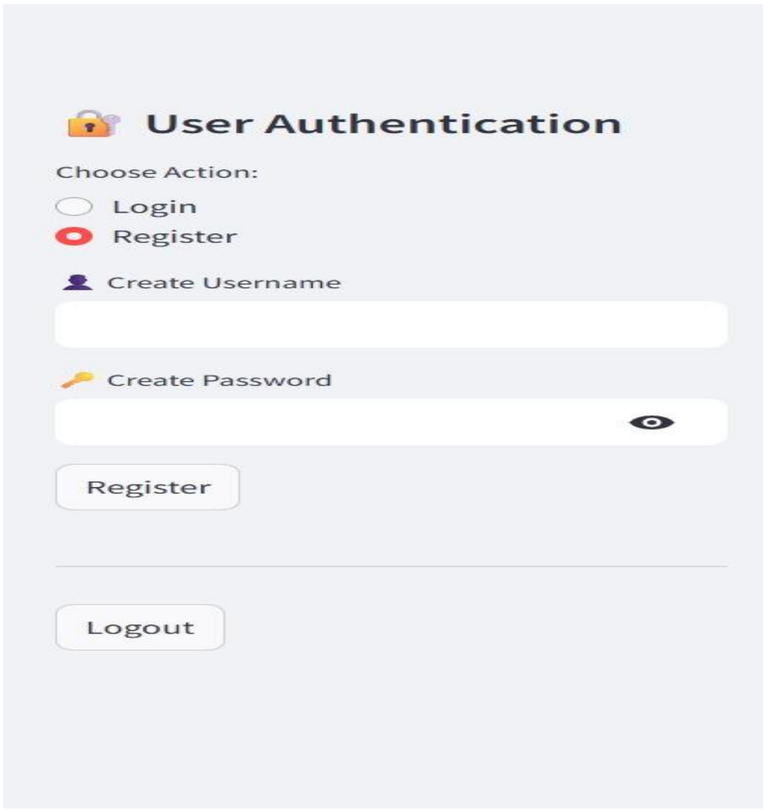
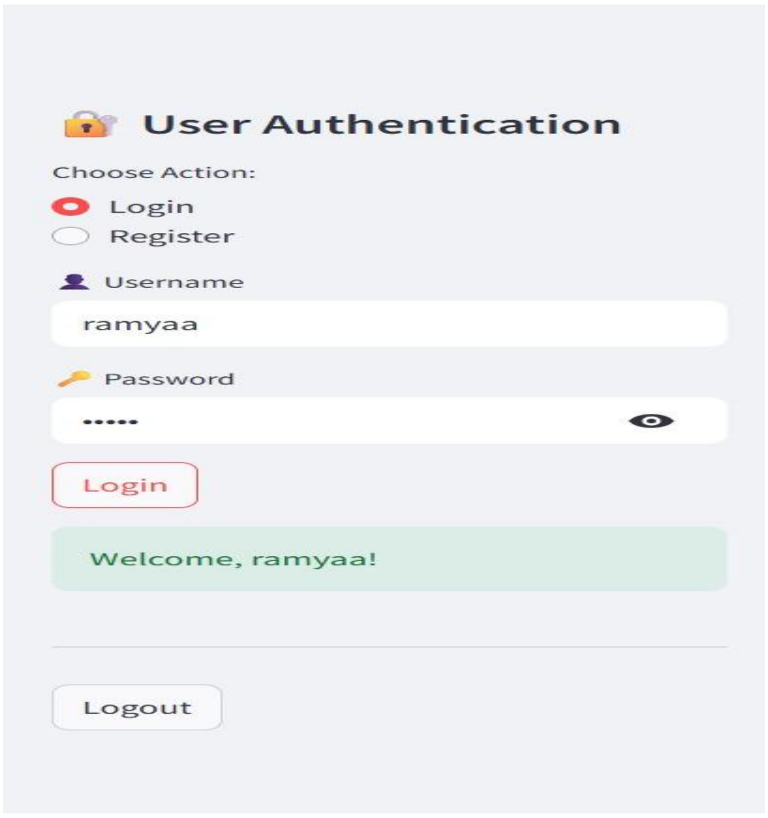


Fig 5.2

5.1.3 USER INTERFACE – LOGIN PAGE

This image shows the **Login Interface** of your Student Performance Predictor System. Here, the user selects the **Login** option, enters a valid **username** ("ramyaa") and **password**, and clicks the **Login** button. Upon successful login, a green confirmation message appears: *"Welcome, ramyaa!"* The interface also includes a password visibility toggle and a **Logout** button for secure session termination. The layout is simple and user-friendly, ensuring smooth authentication for students and educators alike.


User Authentication

Choose Action:

(●) Login
( ) Register

Username
ramyaa

Password
•••••

Login

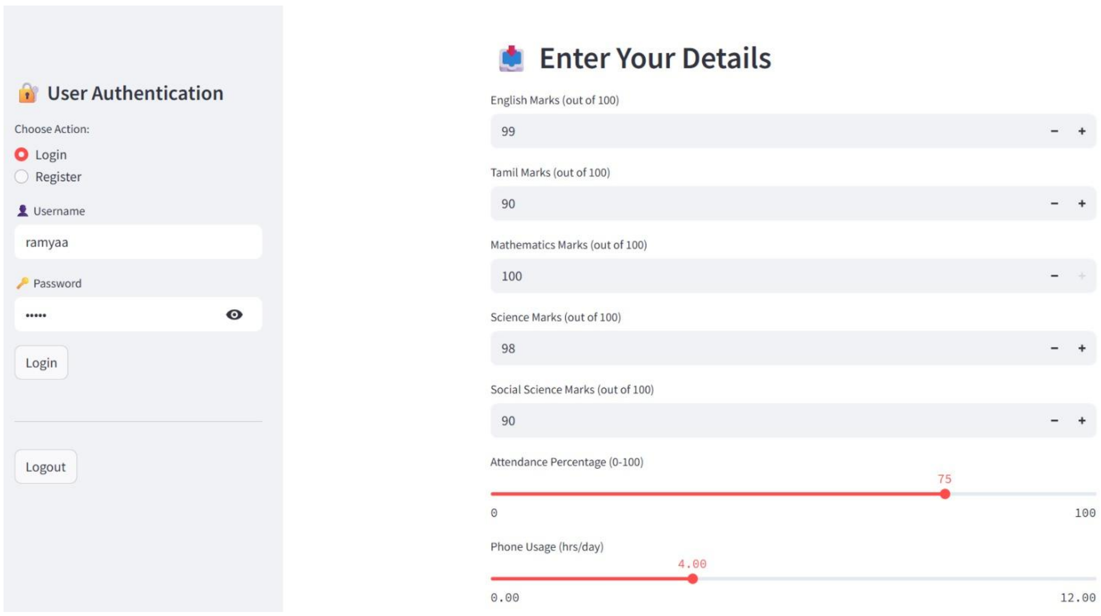
Welcome, ramyaa!

Logout

Fig 5.3

5.1.3 USER INTERFACE – ENTERING STUDENT DETAILS PAGE

This image displays the **Student Input Interface** of your Student Performance Predictor System. After logging in, users can enter their subject-wise marks (English, Tamil, Mathematics, Science, and Social Science), attendance percentage, and daily phone usage. Marks are inputted using numeric fields, while attendance and phone usage are adjusted through sliders for ease of use. This interface captures both academic and behavioral data, enabling the system to predict performance and provide tailored feedback. The layout is clean and intuitive, supporting accurate and quick data entry for students.



User Authentication

Choose Action:

[✓ Login]
Register

Username
ramyaa

Password
•••••

Login

Logout

Enter Your Details

English Marks (out of 100): 99

Tamil Marks (out of 100): 90

Mathematics Marks (out of 100): 100

Science Marks (out of 100): 98

Social Science Marks (out of 100): 90

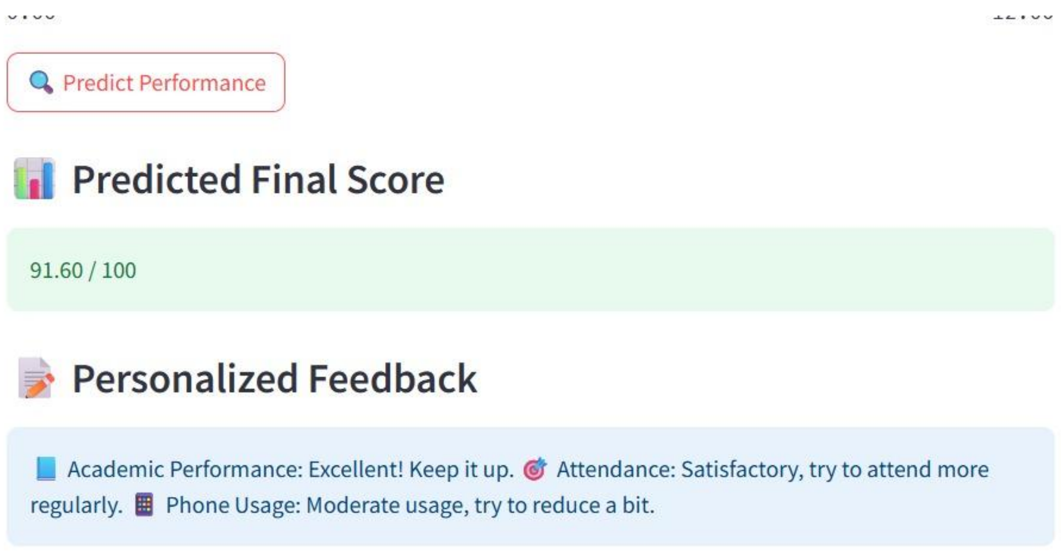
Attendance Percentage (0-100): 75 (0 – 100)

Phone Usage (hrs/day): 4.00 (0.00 – 12.00)

Fig 5.4

5.1.3 USER INTERFACE – PERFORMANCE PREDICTOR

This image shows the Prediction and Feedback Interface of your Student Performance Predictor System. After the user clicks the Predict Performance button, the system displays the predicted final score (e.g., 91.60 out of 100) based on the provided academic and behavioral data. Below the score, personalized feedback is given in three categories: academic performance, attendance, and phone usage. This tailored guidance helps students understand their strengths and areas for improvement, encouraging better study habits and time management**.**



📊 Predicted Final Score

91.60 / 100

📝 Personalized Feedback

📘 Academic Performance: Excellent! Keep it up. 🎯 Attendance: Satisfactory, try to attend more regularly. 📱 Phone Usage: Moderate usage, try to reduce a bit.

Fig 5.5

CHAPTER 6

CONCLUSION AND FUTURE WORK

This project successfully implemented an intelligent, web-based **Student Performance Predictor System** that uses machine learning to forecast student outcomes based on academic and behavioral parameters. By analyzing subject-wise marks, attendance percentage, and phone usage hours, the system accurately predicts a student's final performance score using a trained **linear regression model**. The application offers a secure, user-friendly interface that supports personalized input and delivers real-time predictions along with actionable feedback. Its lightweight design ensures quick processing with minimal computational resources, making it accessible for students, parents, and educators. The integration of frontend and backend components allows for smooth operation, and the modular structure supports scalability.

However, the system has limitations, including dependency on dataset quality and the simplicity of the linear model, which may affect performance in more complex scenarios.

For future work, the system can be enhanced by:

- Expanding the dataset with more diverse and larger samples.
- Incorporating advanced models like **Random Forest** or **Neural Networks** for improved prediction accuracy.
- Including more behavioral parameters such as study hours or sleep patterns.
- Adding data visualization dashboards for better performance tracking.
- Enabling mobile app support for broader accessibility.

Overall, this project demonstrates the potential of combining academic and behavioral analytics for educational improvement and lays a solid foundation for more advanced student assessment tools.

APPENDIX

SOURCE CODE

MLLL.PY

```
import streamlit as st
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import r2_score, mean_squared_error

 if 'users' not in st.session_state:
    st.session_state.users = {'admin': 'admin123'}

if 'logged_in' not in st.session_state:
    st.session_state.logged_in = False

if 'current_user' not in st.session_state:
    st.session_state.current_user = None

 # Sidebar - Login/Register
 st.sidebar.title("User Authentication")

auth_mode = st.sidebar.radio("Choose Action:", ["Login", "Register"])

if auth_mode == "Register":
    new_user = st.sidebar.text_input(" Create Username")
    new_pass = st.sidebar.text_input(" Create Password", type="password")
    if st.sidebar.button("Register"):
        if new_user in st.session_state.users:
            st.sidebar.warning("Username already exists.")
        elif new_user and new_pass:
            st.session_state.users[new_user] = new_pass
            st.sidebar.success("Registration successful! You can now log in.")
        else:
            st.sidebar.warning("Fill all fields to register.")

elif auth_mode == "Login":
    username = st.sidebar.text_input(" Username")
    password = st.sidebar.text_input(" Password", type="password")
    if st.sidebar.button("Login"):
        if username in st.session_state.users and st.session_state.users[username] == password:
            st.session_state.logged_in = True
            st.session_state.current_user = username
            st.sidebar.success(f"Welcome, {username}!")
        else:
            st.sidebar.error("Invalid credentials")

# ---------------------------
# If Logged In: Show Main App
# ---------------------------
if st.session_state.logged_in:

    st.title(" Student Performance Predictor")
    st.markdown("This app predicts student performance using a trained *Machine Learning Model* and gives *smart feedback*.")

    try:
        # Load dataset
        df = pd.read_csv('student_performance_dataset.csv')

        # Split data
        X = df[['academic_score', 'attendance', 'phone_usage']]
        y = df['final_score']
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

        # Train models
        lr_model = LinearRegression()
        lr_model.fit(X_train, y_train)

        ridge_model = Ridge(alpha=1.0)
        ridge_model.fit(X_train, y_train)

        lasso_model = Lasso(alpha=0.1)
        lasso_model.fit(X_train, y_train)

        # Evaluate models
        models = {'Linear Regression': lr_model, 'Ridge Regression': ridge_model, 'Lasso Regression': lasso_model}
        best_model = None
        best_r2 = -np.inf
        for name, model in models.items():
            y_pred = model.predict(X_test)
            r2 = r2_score(y_test, y_pred)
            rmse = np.sqrt(mean_squared_error(y_test, y_pred))
            st.write(f"{name} → R²: {r2:.2f}, RMSE: {rmse:.2f}")
            if r2 > best_r2:
                best_r2 = r2
                best_model = model

        st.success(f" Best model selected: {type(best_model)._name_} (R²:
{best_r2:.2f})")

        st.header(" Enter Your Details")

        english = st.number_input("English Marks (out of 100)", 0, 100, 0)
        tamil = st.number_input("Tamil Marks (out of 100)", 0, 100, 0)
        math = st.number_input("Mathematics Marks (out of 100)", 0, 100, 0)
        science = st.number_input("Science Marks (out of 100)", 0, 100, 0)
        social = st.number_input("Social Science Marks (out of 100)", 0, 100,
0)

        attendance = st.slider("Attendance Percentage (0-100)", 0, 100, 75)
        phone_usage = st.slider("Phone Usage (hrs/day)", 0.0, 12.0, 4.0)

        academic_score = (english + tamil + math + science + social) / 5

        if st.button(" Predict Performance"):
            input_data    =    pd.DataFrame([[academic_score,    attendance,
phone_usage]],
                          columns=['academic_score',           'attendance',
'phone_usage'])
            try:
                predicted_score = best_model.predict(input_data)[0]
                predicted_score = max(0, min(predicted_score, 100))

                st.subheader("  Predicted Final Score")
                st.success(f"{predicted_score:.2f} / 100")

                st.subheader(" Personalized Feedback")
                feedback = ""

                if academic_score >= 85:
                    feedback += "  Academic Performance: Excellent! Keep it
```

```
up.\n"
                elif academic_score >= 70:
                    feedback += "  Academic Performance: Good, but there's
room for improvement.\n"
                else:
                    feedback += " Academic Performance: Needs improvement.
Focus on studies.\n"

                if attendance >= 90:
                    feedback += "Attendance: Great job maintaining high
attendance!\n"
                elif attendance >= 75:
                    feedback += "Attendance: Satisfactory, try to attend more
regularly.\n"
                else:
                    feedback += " Attendance: Low attendance, which may affect
performance.\n"

                if phone_usage <= 2:
                    feedback += " Phone Usage: Excellent self-control!\n"
                elif phone_usage <= 5:
                    feedback += " Phone Usage: Moderate usage, try to reduce a
bit.\n"
                else:
                    feedback += "Phone Usage: Too much phone usage!
Consider cutting down.\n"

                st.info(feedback)

            except Exception as e:
                st.error(f"Prediction error: {e}")

    except FileNotFoundError:
        st.error("⚠ Dataset file 'student_performance_dataset.csv' not found.
Please upload it to the app directory.")
    except Exception as e:
        st.error(f"An unexpected error occurred: {e}")

    st.sidebar.markdown("---")
    if st.sidebar.button("Logout"):
        st.session_state.logged_in = False
        st.session_state.current_user = None
        st.experimental_rerun()
```

```
else:
    st.warning(" Please login or register from the sidebar to continue.")
```

This the source code of the project studenction performance prediction analysis
Using Linear Regression

REFFERENCES

1. M. M. E. Khoudier et al., "Prediction of Student Performance Using Machine Learning Techniques," in *Proc. 2023 5th Novel Intelligent and Leading Emerging Sciences Conf. (NILES)*, 2023, pp. 1–6, doi: 10.1109/NILES59815.2023.10296766.ResearchGate
2. Z. Gupta and S. Gupta, "Monitoring and Predicting Performance of Students in Degree Programs Using Machine Learning," in *Proc. 2023 10th Int. Conf. on Computing for Sustainable Global Development (INDIACom)*, 2023, pp. 1311–1315.ResearchGate
3. B. Althaph, S. V. N. Sreenivasu, and D. V. Reddy, "Student Performance Analysis with Ensemble Progressive Prediction," in *Proc. 2023 5th Int. Conf. on Smart Systems and Inventive Technology (ICSSIT)*, 2023, pp. 1513–1517.ResearchGate
4. F. Ouatik, M. Erritali, F. Ouatik, and M. Jourhmane, "Predicting Student Success Using Big Data and Machine Learning Algorithms," *Int. J. Emerg. Technol. Learn.*, vol. 17, no. 12, pp. 236–245, 2022.ResearchGate
5. H. Pallathadka et al., "Classification and Prediction of Student Performance Data Using Various Machine Learning Algorithms," *Mater. Today Proc.*, vol. 80, pp. 3782–3785, 2023.ResearchGate
6. I. A. A. Amra and A. Y. Maghari, "Students Performance Prediction Using KNN and Naïve Bayesian," in *Proc. 2017 8th Int. Conf. on Information Technology (ICIT)*, 2017, pp. 909–913.ResearchGate
7. R. S. Kumar and J. Kumar, "Analysis of Student Performance Based on Classification and Map Reduce Approach in Big Data," *Int. J. Pure Appl. Math.*, vol. 118, no. 14, pp. 141–148, 2018.ResearchGate
8. S. Rebai, F. Ben Yahia, and H. Essid, "A Graphically Based Machine Learning Approach to Predict Secondary Schools Performance in Tunisia," *Socioecon. Plann. Sci.*, vol. 70, p. 100724, 2020.ResearchGate
9. Z. Ahmad and E. Shahzadi, "Prediction of Students' Academic Performance Using Artificial Neural Network," *Bull. Educ. Res.*, vol. 40, no. 3, pp. 157–164, 2018.ResearchGate

10. X. Xu, J. Wang, H. Peng, and R. Wu, "Prediction of Academic Performance Associated with Internet Usage Behaviors Using Machine Learning Algorithms," *Comput. Human Behav.*, vol. 98, pp. 166–173, 2019.ResearchGate
11. M. L. Bernacki, M. M. Chavez, and P. M. Uesbeck, "Predicting Achievement and Providing Support Before STEM Majors Begin to Fail," *Comput. Educ.*, vol. 145, p. 103728, 2020.ResearchGate
12. Y. Wang, A. Ding, K. Guan, S. Wu, and Y. Du, "Graph-Based Ensemble Machine Learning for Student Performance Prediction," *arXiv preprint arXiv:2112.07893*, 2021.arXiv
13. Y.-W. Chu et al., "Mitigating Biases in Student Performance Prediction via Attention-Based Personalized Federated Learning," *arXiv preprint arXiv:2208.01182*, 2022.arXiv
14. S. Minn, "BKT-LSTM: Efficient Student Modeling for Knowledge Tracing and Student Performance Prediction," *arXiv preprint arXiv:2012.12218*, 2020.arXiv
15. Q. Liu et al., "EKT: Exercise-Aware Knowledge Tracing for Student Performance Prediction," *arXiv preprint arXiv:1906.05658*, 2019.arXiv