

Project Title	AI-Powered Personalized Learning Assistant for School and College Students
Skills take away From This Project	Python (TensorFlow, PyTorch, Scikit-Learn, Hugging Face Transformers) SQL/NoSQL Databases for storing policyholder data AWS/Azure/GCP for cloud deployment Streamlit/Flask for interactive UI
Domain	Education

Problem Statement:

In today's education system, students have diverse learning speeds, styles, and preferences. Traditional teaching methods often follow a "one-size-fits-all" model, which leaves slow learners behind and fails to challenge fast learners. Teachers are overwhelmed with managing large classrooms and cannot provide individual attention. Moreover, students struggle to revise concepts effectively and lack personalized guidance outside school hours. Current EdTech platforms offer video content and static quizzes but do not adapt dynamically to student behavior or performance.

There is a growing need for an AI-driven solution that can:




- Continuously assess student performance,
- Adapt content delivery accordingly,
- Generate practice questions, summaries, and explanations in real-time,
- Offer insights to educators about student progress and potential issues.


By integrating ML for performance prediction, DL for content classification, and LLMs for natural language generation, we can build a truly adaptive learning assistant that enhances both teaching efficiency and student learning outcomes.


Business Use Cases:


- **EdTech companies** can integrate the solution into their LMS to provide customized learning paths.
- **Schools and colleges** can use the assistant for remedial coaching and curriculum support.
- **Parents** gain insights into their child's learning behavior and can take early action.


Revenue potential: Subscription-based models, B2B licensing for institutions, and value-added services like mock test generation.

No	Title	Use Case
1	 Predict Student Pass/Fail	Can you build a classification model to predict whether a student will pass or fail a quiz or exam based on their historical activity, time spent, and past scores? Use logistic regression to classify outcomes and explain the key influencing features.
2	 Score Range Prediction	Can you build a regression model that predicts a student's future score or grade (0-100) using past performance data, topic difficulty, and time spent per question? Use Random Forest Regressor and compare with Linear Regression.
3	 Learning Style Clustering	Can you cluster students into groups like 'visual learners', 'slow learners', and 'fast responders' based on their interaction data, reading speed, and topic-wise performance? Use K-Means and evaluate using silhouette score.



- 4  Dropout Risk Detection


Can you build a model that predicts which students are likely to drop out or stop using the platform based on inactivity, poor performance, or inconsistent engagement? Use XGBoost Classifier and analyze model importance.
- 5  Topic Detection from Student Answers

Can you use a deep learning NLP model to detect the topic or subject area of a paragraph written by a student? Implement LSTM or BiLSTM on labeled textual input and show how accurately it classifies.
- 6  Handwritten Digit Recognition

Can you build a CNN model to classify handwritten digits submitted by students in math assignments? Train a CNN on the MNIST dataset or real digit images scanned from notes.
- 7  AI-Based Topic Summarizer

Can you use a Large Language Model (LLM) to generate a simplified 5-line summary of a long educational article or lesson in your own words? Use GPT or Hugging Face transformer models for text summarization.

No.	Model / Use Case	Dataset	Metadata Fields	Dataset URL
1	 Predict Pass/Fail	ASSISTments 2009-2010 Dataset	student_id, skill_id, problem_id, correct, time_sec, attempt_count, hint_count	https://sites.google.com/view/assistmentsdatamining
2	 Predict Score Range	EdNet Dataset	user_id, timestamp, problem_id, elapsed_time, correct, tag, bundle_id	https://github.com/rriid/ednet
3	 Learning Style Clustering	EdNet or Khan Academy Logs	student_id, time_on_top, ic, avg_score, num_attempts, response_time,	https://github.com/rriid/ednet https://api.khanacademy.org/

			content_type e_viewed	
4	 Dropout Risk Detection	KDD Cup 2015 (MOOC) or Harvard X MOOC Dataset	user_id, video_watch _count, forum_posts , problem_att empts, last_active , dropout_fla g	https://www.kaggle.com/c/kkbox-music-recommendation-challenge https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/29566
5	 Topic Detection (DL-NLP)	Automated Essay Scoring Dataset	essay_id, text, topic, grade	https://www.kaggle.com/datasets/ghoshanurag03/automated-essay-scoring
6	 Digit Recognition (DL-Vision)	MNIST Handwritten Digits (CSV)	image (28x28), label (0–9)	https://www.kaggle.com/datasets/oddrationalale/mnist-in-csv
7	 AI Topic Summarizer	Wikipedia Articles / TED Talk	title, text, section, source_link	https://www.mediawiki.org/wiki/API:Main_page https://www.kaggle.com/datasets/rounakbanik/ted-talks

Transcripts

Project Evaluation metrics:

1 Model Performance Metrics (AI & ML Effectiveness)

● Classification & Regression Models (Risk Classification & Claim Prediction)

- ✓ **Accuracy** – Measures how well the model correctly classifies risk categories.
- ✓ **Precision & Recall (F1-score)** – Balances false positives and false negatives in fraud detection and risk classification.
- ✓ **Mean Absolute Error (MAE) & Root Mean Square Error (RMSE)** – Evaluates the accuracy of claim amount predictions.
- ✓ **AUC-ROC Score** – Measures the model's ability to distinguish between fraudulent and genuine claims.

● Clustering Models (Customer Segmentation)

- ✓ **Silhouette Score** – Measures how well clusters are formed and separated.
- ✓ **Davies-Bouldin Index** – Lower values indicate better clustering quality.

● Fraud Detection Models (Anomaly Detection & Association Rules)

- ✓ **False Positive Rate (FPR) & False Negative Rate (FNR)** – Ensures fraudulent claims are caught while minimizing false alarms.
- ✓ **Precision-Recall Curve** – Evaluates fraud detection effectiveness.

● NLP Models (Sentiment Analysis, Translation, Summarization, Chatbot)

- ✓ **BLEU Score & ROUGE Score** – Measures translation and text summarization accuracy.
- ✓ **Sentiment Analysis Accuracy** – Evaluates correctness in detecting customer sentiment (Positive, Negative, Neutral).
- ✓ **Chatbot Response Time & Relevance Score** – Measures chatbot effectiveness in resolving queries.
- ✓ **Source Code Should Include:**
 - **Data Preprocessing Scripts** (Handling missing values, feature engineering, scaling).
 - **ML Model Training Scripts** (Supervised, Unsupervised, NLP).
 - **Evaluation Metrics Calculations** (Accuracy, AUC-ROC, F1-score, Silhouette Score, etc.).
 - **API/Deployment Code** (Flask, FastAPI, Streamlit for interactive dashboard).
 - **Automated Fraud Detection & Customer Segmentation Modules.**

2 Documentation & Reports

📌 **Comprehensive Project Documentation** (📄 PDF, Markdown, or Notion)

✓ **README.md (GitHub Submission Required)**

- **Project Overview**
- **Dataset Used** (Source, preprocessing details)
- **Machine Learning Techniques Applied**
- **Deployment Instructions**

✓ **Technical Report (PDF or Notion Document)**

- **Executive Summary** – Overview of the problem and proposed AI solution.
- **Exploratory Data Analysis (EDA)** – Insights from visualizations & statistics.
- **Model Training & Evaluation** – Performance comparison of different models.
- **Challenges Faced & Improvements** – Any issues faced and how they were solved.
- **Future Enhancements** – Potential improvements or extensions.

✓ **Presentation Deck (PowerPoint or Google Slides)**

- Problem Statement & Business Use Case
 - Data Insights & Key Findings
 - ML/NLP Model Approaches
 - Real-World Applications & Deployment
-

3 Model Artifacts & Evaluation Results

Trained ML & NLP Models

- Models saved in **Pickle (.pkl)**, **ONNX**, **TensorFlow**, **PyTorch (.pt)** formats.
- Versioning handled using **MLflow** or **GitHub releases**.

Model Performance Metrics & Comparisons

- Tables comparing different ML models on evaluation metrics.
- Charts (Confusion Matrix, AUC-ROC, Precision-Recall Curve).
- Sentiment analysis model accuracy using **BLEU**, **ROUGE**, or **F1-score**.

Feature Importance Analysis

- Explainability techniques like **SHAP**, **LIME** for model interpretability.
-

4 Deployment Files (Optional But Recommended)

API Development (Flask/FastAPI/Web App)

- **Deployed API** to predict claim risk, detect fraud, or summarize policy documents.
- **Live Demo Link** (if deployed on **AWS**, **GCP**, or **Heroku**).

Interactive Dashboard (Streamlit/Gradio/Flask UI)

- Real-time visualization of **risk scores**, **fraud detection results**, **customer segmentation**.
 - Multilingual chatbot integration for insurance query resolution.
-

5 Video Demo

📌 5-10 min project walkthrough video explaining:

- Problem Statement & Approach
- Model Training Process
- Results & Business Impact
- Live Demo of API or Dashboard

📢 Final Checklist for Submission

- ✓ Source Code with Clear Documentation
- ✓ Well-Written Technical Report (Markdown or PDF)
- ✓ Model Artifacts & Evaluation Results
- ✓ Presentation Deck
- ✓ Deployment Files (if applicable)
- ✓ Video Demo (Bonus for better grading/evaluation)

Project Guidelines:

✓ General Coding Guidelines

- Follow **PEP 8 (Python Enhancement Proposal 8)** for consistent formatting.
- Use **meaningful variable names** and avoid one-letter variables (e.g., `customer_age` instead of `ca`).
- Write **modular code** with functions and classes instead of long scripts.
- **Avoid hardcoding values** – use **config files** (`config.yaml`, `.env`).
- Use **logging instead of print statements** (`logging` module in Python).

Project Structure 👍

personalized-learning-assistant/

|

└─ data/

```
| |— raw/      # Raw dataset files
| |— processed/ # Cleaned, merged data
|
|— models/
| |— ml_models/  # Student clustering, performance prediction
| |— llm_outputs/ # Quiz, summary generation
|
|— notebooks/
| |— EDA.ipynb
| |— ML_Models.ipynb
| |— LLM_Integration.ipynb
|
|— src/
| |— data_preprocessing.py
| |— feature_engineering.py
| |— model_training.py
| |— quiz_generator.py
| |— summary_generator.py
|
|— app/
| |— streamlit_app.py
|
|— requirements.txt
```

— README.md

Timeline:

10 days

PROJECT DOUBT CLARIFICATION SESSION (PROJECT AND CLASS DOUBTS)

About Session: The Project Doubt Clarification Session is a helpful resource for resolving questions and concerns about projects and class topics. It provides support in understanding project requirements, addressing code issues, and clarifying class concepts. The session aims to enhance comprehension and provide guidance to overcome challenges effectively.

Note: Book the slot at least before 12:00 Pm on the same day

Timing: Monday-Saturday (4:00PM to 5:00PM)

Booking link : <https://forms.gle/XC553oSbMJ2Gcfug9>

For DE/BADM project/class topic doubt slot clarification session:

Booking link : <https://forms.gle/NtkQ4UV9cBV7Ac3C8>

Session timing:

For DE: 04:00 pm to 5:00 pm every saturday

For BADM 05:00 to 07:00 pm every saturday

LIVE EVALUATION SESSION (CAPSTONE AND FINAL PROJECT)

About Session: The Live Evaluation Session for Capstone and Final Projects allows participants to showcase their projects and receive real-time feedback for improvement. It assesses project quality and provides an opportunity for discussion and evaluation.

Note: This form will Open only on Saturday (after 2 PM) and Sunday on Every Week

Timing:

For BADM and DE

Monday-Saturday (11:30AM to 1:00PM)

For DS and AIML

Monday-Saturday (05:30PM to 07:00PM)

Booking link : <https://forms.gle/1m2Gsro41fLtZurRA>