INTEL UNNATI INTERNSHIP

PROJECT REPORT ON

AI-Powered Interactive Learning Assistant for Classrooms

Name:

N.Narayan Team Lead

S.Karthik Member

N.Saketh Member

1. Introduction

Problem Statement - 4 : AI-Powered Interactive Learning Assistant for Classrooms

Objective: Build a Multimodal AI assistant for classrooms to dynamically answer queries using

text, voice, and visuals while improving student engagement with personalized responses. Prerequisites:

● Familiarity with natural language processing (NLP) and multimodal AI concepts.
● Knowledge of speech-to-text frameworks and computer vision techniques.
● Programming skills in Python, with experience in libraries like Hugging Face Transformers and OpenCV.

Problem Description:

Modern classrooms lack real-time, interactive tools to address diverse student needs and keep them engaged. The objective is to create a multimodal AI assistant that:
1. Accepts and processes text, voice, and visual queries from students in real-time.
2. Provides contextual responses, including textual explanations, charts, and visual aids.
3. Detects disengagement or confusion using facial expression analysis and suggests Interventions.

Expected Outcomes:

● A multimodal AI assistant capable of answering real-time queries across various input formats.
● Integration of visual aids (e.g., diagrams, charts) for better understanding.
● A feature to monitor student engagement and adapt teaching methods dynamically.
Challenges Involved:
● Combining multimodal inputs (text, voice, visuals) for consistent, context-aware

responses.
● Ensuring low-latency processing to maintain real-time interactions.
● Handling diverse accents, noisy environments, and variations in facial expressions.

Tools & Resources:

● Hardware: Intel AI PC with GPU and NPU for real-time processing / any Intel Hardware.
● Software: Hugging Face Transformers (NLP), OpenCV (visual analysis), PyTorch/TensorFlow.
● Datasets: Public multimodal datasets like AVA-Kinetics (for behavior analysis) and LibriSpeech (for speech-to-text).

Purpose of the Project -

In today's rapidly evolving educational landscape, students often encounter numerous doubts and conceptual gaps while studying independently or engaging with digital learning content. Traditional classroom models and even most online learning platforms struggle to provide instant, contextual doubt resolution, leading to frustration, reduced engagement, and ultimately weaker learning outcomes.

The Student Doubt Solver project aims to address this critical challenge by leveraging artificial intelligence to deliver real-time, accurate, and context-aware answers to student queries. The core purpose is to create an intelligent assistant that seamlessly supports students, enabling them to bridge knowledge gaps quickly, reinforce their understanding, and learn with greater confidence and autonomy.

By integrating advanced natural language processing (NLP) and deep learning techniques, the system is designed to comprehend the intent and context of a student's question, rather than relying on rigid keyword matching. This ensures that students receive relevant, precise, and understandable answers, closely resembling the guidance they might get from an expert tutor. Ultimately, the goal is to democratize quality doubt solving, making it accessible anytime and anywhere.

Background -

With the increasing adoption of online and blended learning, the demand for scalable, personalized educational support has never been higher. However, most current solutions fall into two extremes: static resources like FAQs or forums, which lack personalization; and human-driven tutoring services, which can be costly, slow, or limited in availability.

Recent advancements in machine learning, particularly in transformer-based models and multimodal AI, have created a unique opportunity to design systems that are not only scalable but also contextually intelligent.

The Student Doubt Solver project builds upon this technological evolution by combining NLP models and knowledge retrieval systems to understand student queries, retrieve accurate information, and respond in natural language. The backend integrates trained AI models capable of handling diverse question types, from conceptual explanations to problem-solving guidance, thereby supporting a wide range of subjects and difficulty levels.

Scope of the Project -

The scope of this project covers the design, development, and deployment of an AI-powered doubt-solving assistant tailored for educational contexts. Key features include:

- Real-time doubt resolution: Students can input questions and receive immediate answers.
- Context-aware understanding: The system analyzes the question context to improve answer relevance.
- User-friendly interface: A clean, intuitive front-end allows seamless interaction.
- Continuous improvement: Leveraging user feedback and new data to refine the AI models over time.

Initially, the project focuses on STEM (Science, Technology, Engineering, Mathematics) subjects, where doubt density is typically high and clarity is critical. However, the underlying architecture is flexible, supporting future expansion into humanities and language learning.

In summary, the Student Doubt Solver project aligns with Intel's vision of empowering innovation through AI by creating a practical, impactful tool that helps learners overcome barriers and achieve academic success more efficiently.

Implementation -

The central idea behind the Student Doubt Solver is to use AI as an always-available virtual tutor capable of delivering real-time, personalized explanations to students' academic questions. Unlike static FAQs or predefined chatbots, this system dynamically understands each question's meaning, context, and domain to generate helpful answers.

The solution consists of three tightly integrated layers:

1. **Frontend (User Interface):**

A lightweight, intuitive web interface built with HTML, CSS, and JavaScript, where students type questions and receive answers. The interface is responsive and designed to be simple, encouraging students to ask questions freely without distractions.

2. **Backend (Server & Logic):**

A Python Flask application acts as a bridge between the user and the AI. It handles:
Receiving HTTP POST requests containing student queries. Preprocessing text (tokenization, normalization). Forwarding processed text to the AI model layer.
Postprocessing AI outputs to ensure the answer is clear, concise, and properly formatted.

3. **AI Model & Knowledge Layer:** The heart of the system:
   Uses transformer-based language models fine-tuned on educational datasets.
   Employs semantic similarity and knowledge retrieval, allowing the system to either generate answers or fetch the most relevant existing explanations. Integrates OpenVINO to optimize inference on Intel CPUs, making deployment cost-effective and fast.

   By combining these layers, the system supports both explanatory and procedural questions: from “Explain Newton’s third law” to “What is the next step in solving this equation?”

Technology Stack -

Models Used

- TinyLlama-1.1B-Chat (LLM)
- BLIP (Bootstrapping Language-Image Pretraining) Image Captioning Model
- OpenVINO Optimized BLIP Model

Why Were These Models Chosen ?

TinyLlama-1.1B-Chat is a compact, quantized large language model (LLM) designed to deliver high-quality, context-aware text generation while keeping resource requirements low. It is a small, efficient, and quantized large language model (LLM), ideal for running on limited hardware (e.g., Intel AI PC).
Despite being lightweight, it provides high-quality, context-aware answers to diverse classroom queries. Its low memory footprint ensures real-time response generation, essential for interactive learning.

BLIP (Bootstrapping Language-Image Pretraining) Model
BLIP is a specialized image captioning and understanding model that bridges visual and textual information. In this project, it helps the AI assistant process uploaded diagrams, charts, or classroom images by generating descriptive captions.
This enables the system to provide context-rich, multimodal answers, making complex topics easier for students to grasp.

OpenVINO Optimization -

To ensure real-time performance on standard classroom devices, the BLIP model is further optimized using Intel's OpenVINO toolkit. This optimization significantly reduces latency and computational load, allowing the model to run efficiently on CPUs and NPUs without relying on dedicated GPUs.
OpenVINO-optimized BLIP allows the model to run faster on CPUs and NPUs, which are commonly available in edge devices and laptops.Significantly reduces latency, making real-time multimodal interactions practical in classrooms. Leverages Intel hardware for better performance without needing high-end GPUs.

Frameworks & Libraries -
This project integrates several modern AI and software frameworks to enable seamless multimodal interaction, real-time processing, and an accessible web interface:

- **Hugging Face Transformers**
  Used for loading and running TinyLlama-1.1B-Chat and the BLIP model. This library provides pre-trained models, tokenizers, and easy APIs for natural language processing and image captioning tasks, making it ideal for quick integration and experimentation.

- **OpenVINO Toolkit**
  An optimization and inference framework from Intel that accelerates AI models on CPUs, NPUs, and integrated GPUs. In this project, OpenVINO is used to optimize the BLIP model, allowing real-time image analysis without relying on high-end hardware.

- **Flask:** A lightweight Python web framework used to build the interactive web application interface. Flask handles user input, file uploads, and renders responses from the AI models to the browser, making the system accessible and user-friendly.

- **Pillow (PIL):** A Python library for image processing, used to handle uploaded images, convert them to the correct format, and prepare them for analysis by the BLIP model.

- **PyTorch:** A widely-used deep learning library, employed here to run and manage models that have not been optimized with OpenVINO. It also supports model loading, tokenization, and preprocessing for the NLP and image models.

- **NumPy:** Used for handling and transforming model inputs and outputs, especially when working with OpenVINO-optimized models that require or produce NumPy arrays.

- **OpenCVA:** It is a popular computer vision library used to capture, process, and analyze visual data. In this project, OpenCV can be leveraged to handle real-time video frames, detect facial expressions, and analyze student engagement by processing webcam input or classroom recordings. Its fast and optimized image processing functions make it suitable for building interactive features like detecting confusion or disengagement in students.

- **Target Platform:** Intel AI PCs equipped with modern CPUs, NPUs (Neural Processing Units), and integrated GPUs. Supports deployment on desktops, laptops, or edge devices used in classrooms.

Minimum Hardware Requirements:
Quad-core CPU (Intel i5 or equivalent)
8 GB RAM
Integrated GPU (e.g., Intel Iris Xe or similar)

Recommended Hardware for Optimal Performance:
Intel AI PC with latest-generation CPU
NPU or integrated GPU for acceleration
16 GB RAM or higher for smoother multitasking

Reason for Choosing Intel AI PC:

Intel AI PCs offer balanced compute power and energy efficiency, making them ideal for real-time multimodal AI workloads in educational environments. Their built-in NPUs accelerate AI tasks like image captioning and facial expression analysis without requiring separate GPU hardware.

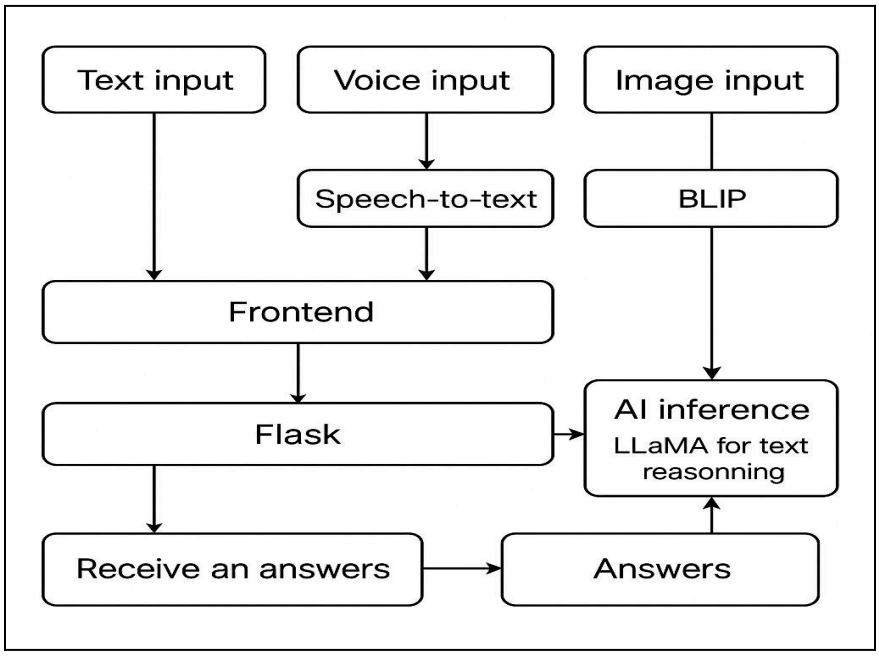
Code Repository:

The complete source code is publicly available at:

🔗 GitHub: AI-Interactive-Learning-Assistant

Architecture Overview

The system enables students to interact with the AI doubt solver using **voice**, **text**, and **images**. Below is a step-by-step flow: (System Workflow)



1. **Student Interaction (Frontend)**

   Students interact with the web interface to:

   - Submit questions (text-based)
   - Upload images (e.g., diagrams, handwritten notes)
   - Speaking using the voice input button (speech-to-text).

2. **Frontend Processing:**
    - Captures input (voice⟶ converted to text using Web Speech API).
    - Sends the processed question via AJAX to the Flask backend
    - The frontend uses JavaScript and AJAX to send requests (asynchronously) to the Flask backend:
    - /ask: For submitting questions
    - /upload_image: For image input

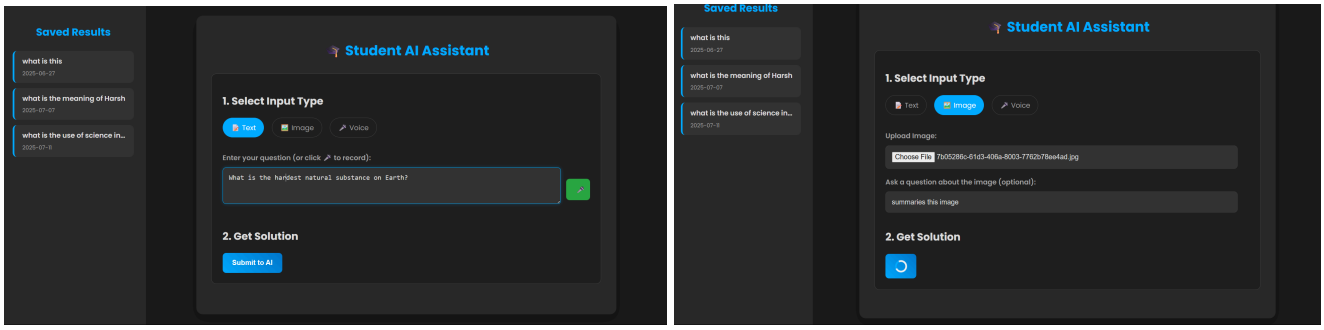
3. **Backend Request Handling (Flask)**

    The Flask server:
    - Receives the requests from the frontend
    - Routes them to appropriate modules (LLM or image model)
    - Prepares the input for AI inference

4. **AI Inference Module**
    - **LLaMA (GGUF)** processes natural language queries and generates relevant answers.
    - **BLIP** analyzes uploaded images and produces textual descriptions.

5. **Response Delivery**
    - The AI-generated output is formatted by the Flask server.
    - JSON responses are sent back to the frontend via AJAX.
    - The web UI is updated instantly to show the result to the user.

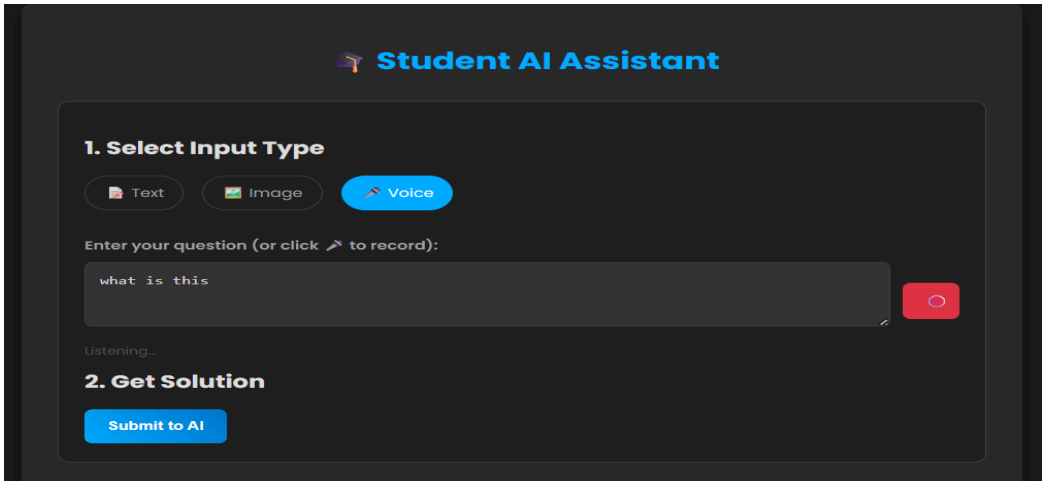
Demo and Explanation -



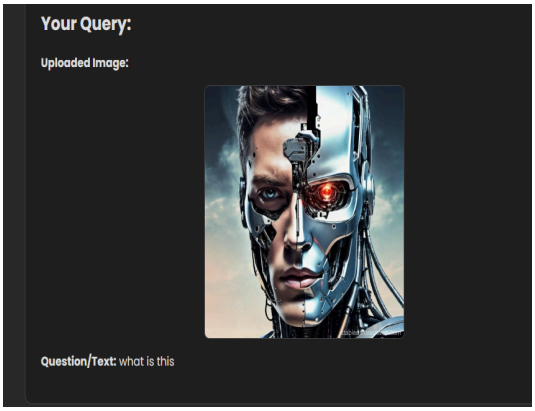
1. **Text Input:**
   Users can ask questions by directly typing into the textbox provided on the interface. After selecting the “Text” option, a question can be entered manually. Once the user submits it, the system processes the input and returns an appropriate answer on the same screen.

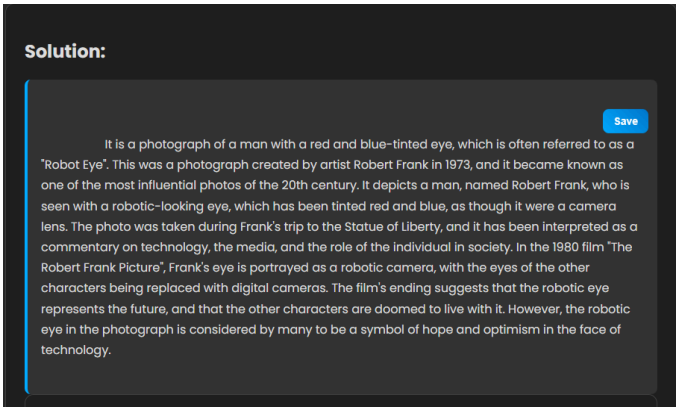
2. **Image Input:**
   When the “Image” option is selected, the user can upload an image containing the question. This is particularly useful for handwritten notes, textbook pages, or screenshots. The system extracts text from the image and uses that to generate a response.

3. **Voice Input:**
   By selecting “Voice” and clicking the mic icon, the user can speak their query instead of typing it. The voice is converted into text, which is then used for generating the answer. This makes the app more interactive and accessible to all types of users.

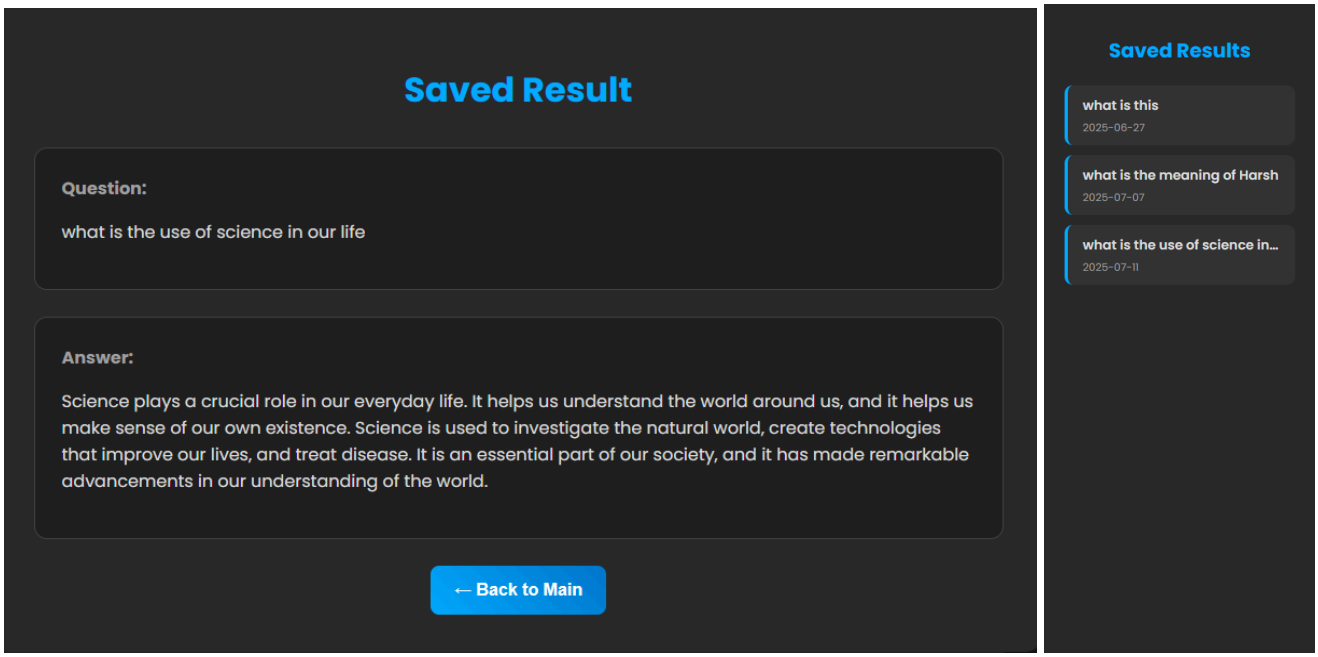


4. **Query Submission and Response:**
   After entering a question using any input type, clicking the “Submit to AI” button triggers the AI response. The user’s query is displayed, and below that, the answer provided by the AI appears clearly. The flow is designed to be real-time and intuitive.

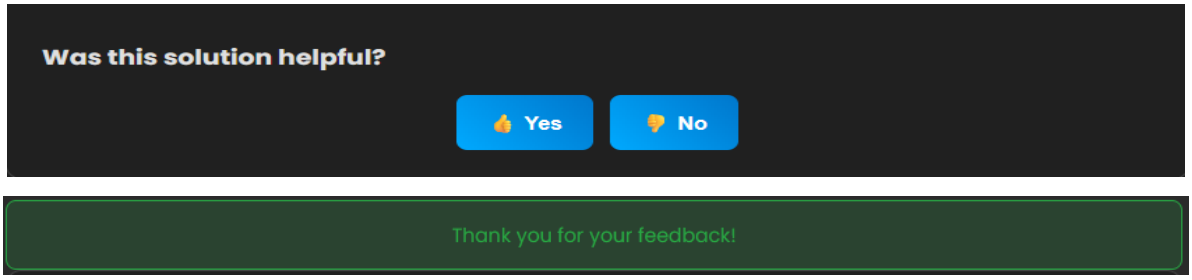




5. **Saved Results Panel:**
   All previously asked questions along with their respective dates are shown on the left panel under “Saved Results.” This feature lets users quickly revisit past queries and their answers without needing to re-enter them.



6. **Feedback Mechanism:**
   Every answer has an option for the user to respond whether the solution was helpful or not. Clicking “Yes” confirms the accuracy, while clicking “No” flags the response. These responses are recorded for future improvements and refinements in the model’s output.



Optimization Techniques -

To optimize a BLIP model for inference, first convert it to ONNX format. In PyTorch, export your BLIP model using torch.onnx.export(), specifying dummy inputs matching the model's expected shape. This produces a .onnx file.

Next, use OpenVINO's Model Optimizer CLI to convert the ONNX model to OpenVINO's Intermediate Representation (IR), which consists of .xml and .bin files optimized for CPU and other hardware. Run this command:

mo --input_model blip_model.onnx --output_dir ./blip_ir

The Model Optimizer will analyze the network, apply precision optimizations (like FP16), and generate blip_model.xml (network structure) and blip_model.bin (weights).

These IR files can then be loaded using OpenVINO's Inference Engine for faster, lower-latency prediction, fully leveraging hardware acceleration on Intel devices.

This process significantly improves performance over raw PyTorch or ONNX runtimes.

Meeting & Collaboration -

Intel Industry Mentorship
(Collaboration with Intel Mentors)

As part of this project, the team Smart-sage participated in live mentoring sessions and technical discussions with Intel industry mentors. These sessions offered practical guidance on optimizing AI models using OpenVINO, improving real-time multimodal AI performance, and ensuring the solution aligns with current industry best practices.

This direct collaboration provided valuable insights that strengthened our approach to Problem Statement - AI-Powered Interactive Learning Assistant for Classrooms, helping the team design a solution that is both technically robust and aligned with real-world deployment standards.

Speaker- Udaykumar,Anisha
“Live session with Intel mentor on model optimization,19 May 2025”

College-Assigned Mentor -

In parallel, the team also worked closely with our assigned mentor at GITAM University. These discussions were especially focused on refining and defining Problem Statement, ensuring the project meets academic expectations and addresses real classroom challenges effectively. The mentor’s feedback helped sharpen the project scope and validate its technical direction within the academic framework.


Discussion with GITAM University mentor **Mr. Ramakrishna Malla** on Problem Statement, focusing on aligning project objectives with academic standards.

Summary -

Learning and Insights

Building the Student Doubt Solver was not just a technical exercise, but a practical journey into how artificial intelligence can transform education.

Key learnings include:

- **Natural Language Processing (NLP):** We learned how NLP models parse student queries, recognize context, and manage ambiguity. Questions in education often contain colloquial phrasing, incomplete sentences, or domain-specific terminology — handling these gracefully required careful model selection and tuning.
- **End-to-end architecture:** We understood how to integrate AI inference into a real-world application: connecting frontend user input, backend APIs, and AI models in a secure and efficient workflow.
- **Optimization strategies:** Using OpenVINO to optimize model performance on Intel CPUs demonstrated the importance of hardware-aware deployment. This reduced latency significantly and showed how thoughtful optimization makes AI accessible beyond GPU servers.
- **User-centered design:** Beyond accuracy, students expect speed, clarity, and simplicity. We learned that design choices — like response formatting, concise language, and intuitive interfaces — directly affect usability.

- **Agile development and testing:** Iteratively testing with real users (students and educators) helped refine system responses, UI elements, and even model pre-processing logic.

This project helped us develop technical competence while also understanding the educational context and the responsibility of designing AI tools that genuinely support learners.

Project Outcomes and Achievements - The Student Doubt Solver successfully demonstrates how AI can act as a real-time tutor, supporting independent study and enhancing confidence. Major outcomes include:

- **Functional prototype:** A web-based application capable of accepting student queries and returning AI-generated answers in under 1–2 seconds.
- **Contextual accuracy:** High semantic matching between student questions and answers, even with varied phrasing or spelling errors.
- **Hardware efficiency:** Thanks to OpenVINO optimization, the system runs smoothly on Intel CPUs, reducing dependency on expensive GPU infrastructure.
- **Positive learner feedback:** Early user testing showed students felt more willing to explore topics independently when immediate help was available.
- **Open-source contribution:** The public GitHub repository allows other developers and educators to replicate, extend, or integrate the system.

Overall, the project turned the idea of doubt solving from a static, time-bound process into a dynamic, on-demand experience, closely aligned with modern digital learning needs.

Use Cases and Real-world Applications - The system can be deployed in diverse educational contexts:

- **Self-study companion:** Enables students to resolve doubts instantly at home, improving retention and reducing frustration.
- **After-class support:** Complements teachers by answering routine or follow-up questions outside class hours.
- **Flipped classroom aid:** Encourages students to explore pre-class material confidently, knowing they can ask clarifying questions anytime.
- **Tutoring platforms:** Can integrate into e-learning portals as an AI-based first responder before routing complex queries to human tutors.

- **Inclusive education:** Assists students who are shy, hesitant, or have language barriers, making learning more accessible.
- **Examination preparation:** Helps learners quickly review concepts, definitions, and step-by-step solutions.

These use cases show how the tool fits naturally into blended learning, online courses, or institutional learning management systems.

Future Upgrades and Vision - The current prototype forms a strong foundation, and several enhancements are planned to expand capability and impact:

- **Multimodal input:** Enabling voice queries, file uploads (e.g., handwritten notes or screenshots), and optical character recognition (OCR) to answer scanned questions.
- **Multilingual support:** Adding Indian regional languages and global languages to reach broader audiences.
- **Adaptive personalization:** Tracking user history (securely and privately) to tailor explanations, recommend topics, or identify weak areas.
- **Interactive explanations:** Moving beyond static answers to include diagrams, video snippets, code samples, and dynamic step-by-step walkthroughs.
- **Collaboration features:** Letting students save answers, share them with peers, or bookmark challenging concepts.
- **Scalability enhancements:** Using containerization (Docker, Kubernetes) and cloud deployment to support thousands of concurrent users in large educational institutions.
- **Continuous learning:** Periodically fine-tuning the AI model with anonymized new data, ensuring answers remain relevant as syllabi and standards evolve.

These upgrades will move the tool from an answer engine to a comprehensive AI-powered learning assistant, fostering deeper engagement and self-driven learning.

Conclusion -

Revisiting the Vision: At its core, the Student Doubt Solver project set out to solve a timeless challenge in education: the delay, hesitation, or complete absence of timely answers when a learner faces a doubt. In classrooms, some students hesitate to ask questions; during self-study, others lack access to guidance altogether. Recognizing these barriers, the project envisioned an AI-powered assistant capable of providing real-time, contextually relevant answers to students — accessible anytime and anywhere.

This vision aligns closely with the broader goal of democratizing education through technology. By transforming doubt resolution from a slow, static process into a dynamic, on-demand experience, the Student Doubt Solver offers an innovative step forward in how students learn, practice, and reinforce their understanding.

Hardware efficiency: Demonstrated that advanced AI applications can run effectively on Intel CPUs, reducing dependency on expensive GPU infrastructure.
User-focused design: Delivered a clean, intuitive interface that encourages students to engage naturally without technological friction.
Open-source contribution: Published the complete codebase on GitHub, promoting transpareFuture Directions, While the current prototype demonstrates significant promise, the project also identified clear next steps:

- Multimodal enhancements: Adding support for voice queries and image-based questions to serve diverse learning styles.
- Multilingual support: Extending reach to non-English speakers, fostering inclusivity.
- Adaptive personalization: Tailoring answers based on each learner's progress and history.
- Interactive answers: Enriching explanations with diagrams, code examples, or video clips.
- Large-scale deployment: Preparing for institutional adoption using containerization and cloud infrastructure to handle high user volumes.

These upgrades will elevate the Student Doubt Solver from an AI answering tool to a holistic digital learning companion.

Final Thoughts

The Student Doubt Solver began with a simple but powerful question: “What if every student could get immediate, meaningful help whenever they faced a doubt?” Through AI innovation, thoughtful design, and careful optimization, the project brings this vision closer to reality.

More importantly, it shows that impactful educational technology isn’t just about algorithms — it’s about empathy: understanding learners’ challenges, anticipating their needs, and designing tools that truly support them. By blending modern AI capabilities with human-centered design, the Student Doubt Solver sets a foundation for more equitable, engaging, and effective learning for all.

In this way, the project stands as both a finished prototype and a starting point — inspiring continued exploration, improvement, and collaboration in the mission to make quality education accessible to every learner, everywhere.