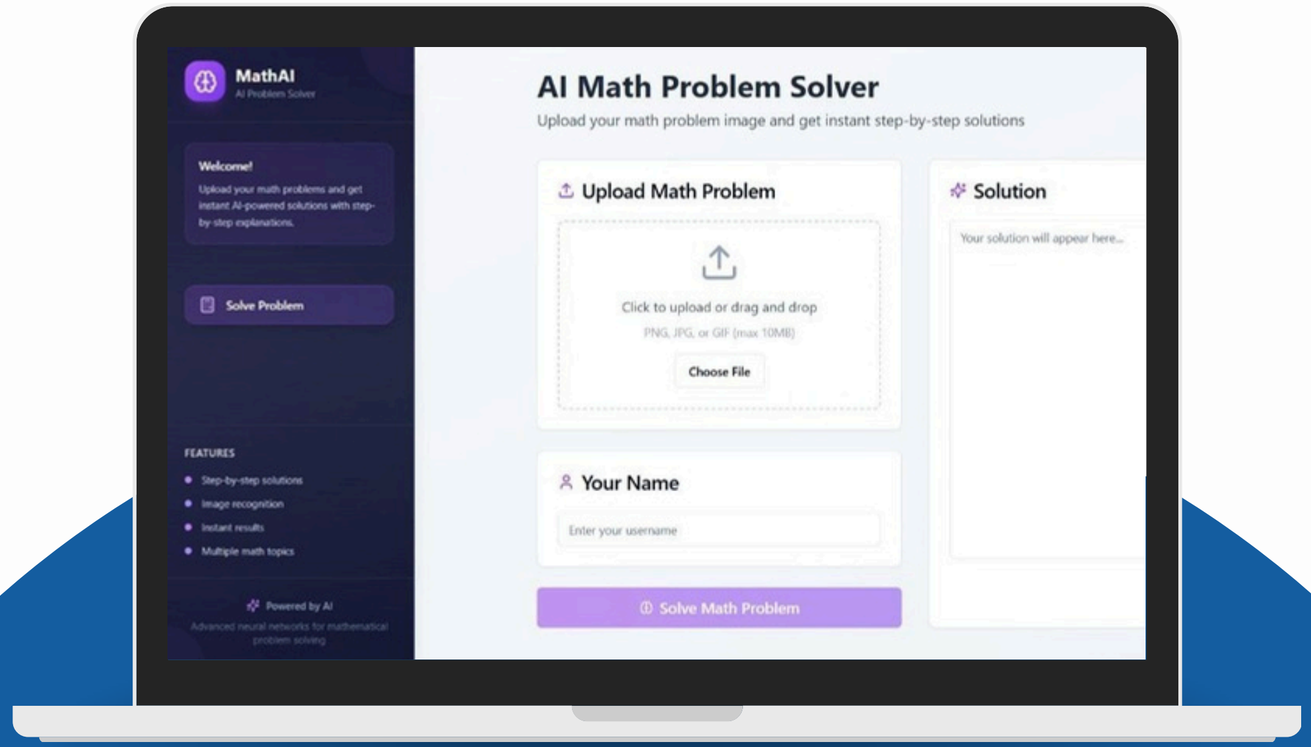
Final Mini-Project Presentation

AI Math Problem Solver



Presented by:
Hasan Chreim 6505
Hussein Rkein 5838

Introduction

Our project **<<MathAI>>** is an AI-powered web application designed to solve mathematical problems from images.

The goal is to provide users with an instant, accurate, and user-friendly math-solving tool.

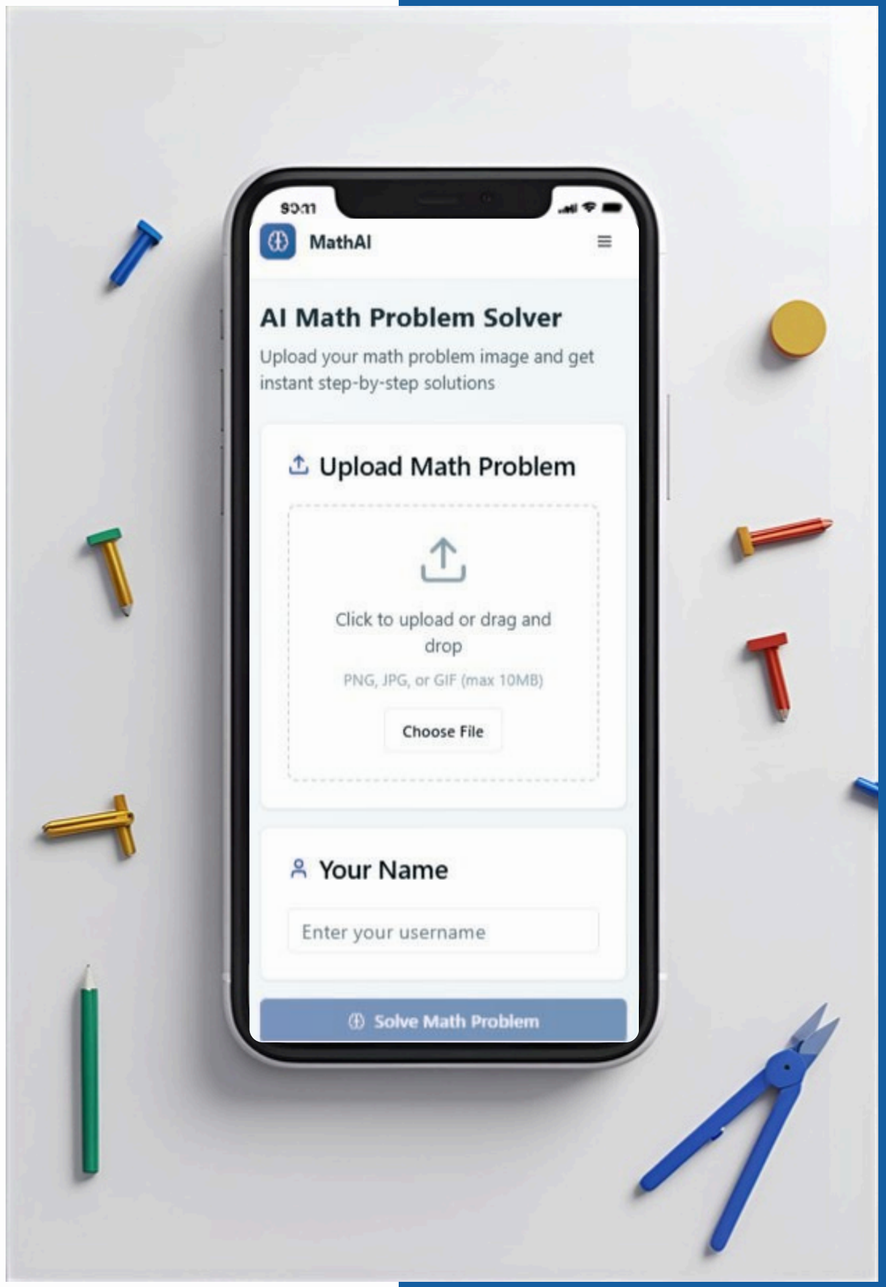
Problem Statement

The idea originated from the need for quick, clear help with math problems. Many tools require manual typing or lack detailed explanations.

We built this app to solve that by letting users snap a photo and get accurate, AI-generated solutions.

Who may use our App

- Students
- Teachers
- Researchers
- Self-Learners



Solution Summary

OCR Image Processing

image is scanned to extract text using Optical Character Recognition (OCR)

Result Display

A clear, step-by-step solution is returned and shown to the user

Image Input

User uploads or captures a photo of a math problem

AI Agent (Mistral)

The extracted text is sent to the AI model to interpret and solve the problem.

Database Logging

Each solution is saved for user history and learning tracking

Key Features

1. **Image-Based Input:**
   Users can upload or capture a photo of a math problem — no need to type manually.

2. **Instant Results with Explanation:**
   AI model solves problems in satisfactory time with good precision, along with step-by-step explanation

3. **Responsive Frontend:**
   Clean and User-Friendly GUI, with responsive frontend to access the app from any device

4. **History & Records:**
   Previous problems and solutions are saved for future review and tracking progress

5. **Multiple Math Topics Supported:**
   Covers arithmetic, algebra, calculus, equations, and more — adaptable to various levels

System Architecture

The app is built using a modular architecture that connects frontend, backend, AI engine, and database for smooth operation:

React

- User uploads an image or takes a photo.
- Displays the solution and history.
- Communicates with the backend via API calls.

Frontend

python™

- Receives image from frontend.
- Handles OCR (image-to-text conversion).
- Sends the math text to the AI engine.
- Returns AI response (solution) to the frontend.
- Logs problem and solution to the database.

Backend

MISTRAL AI_

- Receives parsed math problem as a prompt.
- Processes and returns step-by-step solution using reasoning capabilities.

AI Agent

MySQL®

- Stores user-submitted problems, AI-generated solutions, and timestamps.
- Supports user history and future analytics.

DataBase

Technology Stack

Backend

Flask (Python) — Lightweight web framework to handle API requests, image processing, and logic.

AI Agent

Mistral — A powerful language model used to understand and solve math problems with step-by-step reasoning.

Version Control

GitHub — Version control and collaboration

Frontend

React.js — For building a fast, responsive user interface.

OCR

LATEX-OCR — Extracts math text from uploaded images.

DataBase

MySQL — Stores problems, solutions, and user history for retrieval and tracking.

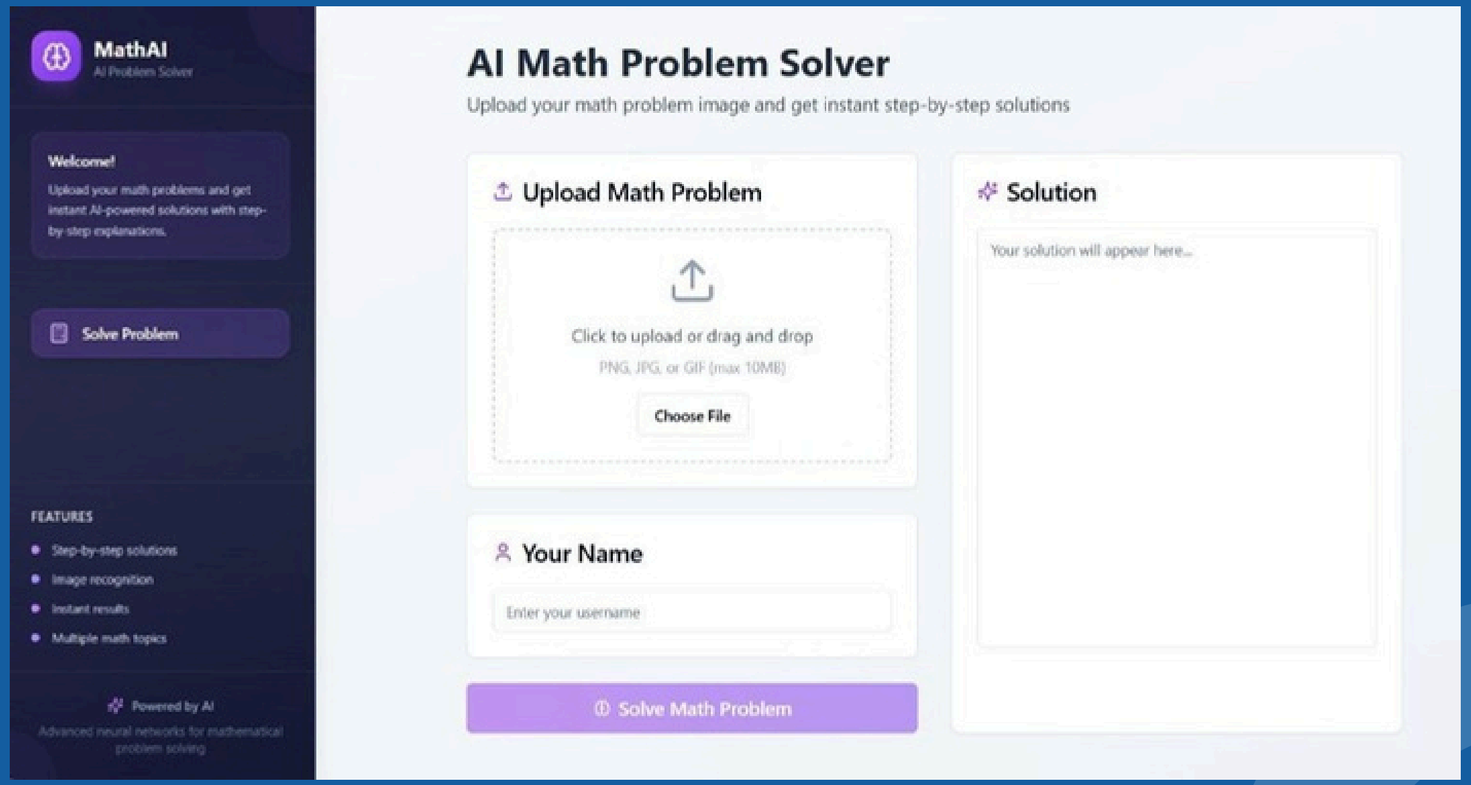
SYSTEM IMPLEMENTATION

Frontend

Our frontend is built using **React.js**, a popular JavaScript library for building dynamic and responsive user interfaces. It allows users to easily upload or capture math problem images and view solutions in real time.

The interface is designed to be clean, intuitive, and mobile-friendly.

The frontend also handles user history display, providing access to previously solved problems for learning and review.



Backend

Our Backend part is built-up of four main sections

App.py

Acts as the backend server

db.py

Manages MySQL connection pooling for efficient DB access

recognition.py

Extracts math expressions from images using LaTeX OCR

solution.py

Sends math problems (LaTeX) to Mistral AI for solving and stores results in DB

```
# ── Initialize a connection pool ──
_____
____
try:
    _connection_pool =
 pooling.MySQLConnectionPool(
        pool_name    = "mypool",
        pool_size    = 5,
        **db_config
    )
except mysql.connector.Error as err:
    if err.errno ==
 errorcode.ER_ACCESS_DENIED_ERROR:
        print(
"❌ DB authentication error: check DB_
USER/DB_PASSWORD"
)
    elif err.errno ==
 errorcode.ER_BAD_DB_ERROR:
        print(
"❌ Database does not exist: check DB_
NAME"
)
    else:
        print(f"❌ MySQL Error: {err}"
)
```

db.py

Code Walkthrough

- **Config:** Loads DB credentials from .env.
- **Pool Setup:** Creates 5 pre-connected DB connections (reduces overhead).
- **Error Handling:** Catches auth/DB-not-found issues during startup.
- **Usage:** get_connection() leases a connection from the pool.

Why Connection Pooling?

- **Pros:** Faster (reuses connections), avoids timeout issues, limits concurrent DB load.
- **Cons:** Slightly higher memory usage, pool size must be tuned.

recognition.py

Without OCR, the app can’t interpret uploaded math images.

Why LaTeX OCR?

- **Pros:** Specialized for math expressions, accurate for printed equations.
- **Cons:** Slower than general OCR, requires model setup.

LATEX OCR

```
    # Step 3: Run OCR and update DB
    latex_code =
extract_math_expression(temp_path)
    if latex_code:
        print("  Recognized LaTeX:",
latex_code)
    else:
        print(
"⚠️ OCR model returned no result.")
    update_recognized_text(image_id,
latex_code)
```

solution.py

Core functionality - without this, app can't solve math problems

- Timeout handling (5s connect, 120s read)
- Clean DB integration

Mistral AI

```
    prompt = (
        f"Given the following LaTeX mathematical expression:\n\n"
        f"{latex_code}\n\n"
        "Please return ONLY the final simplified or evaluated result "
        "(just the answer, no explanation)."
    )

    print(f"📤 Sending to Mistral (model={PRIMARY_MODEL}) for image ID {image_id}...")
    answer = send_to_mistral(prompt)
    if answer is None:
        print("❌ Failed to get a solution from Mistral.")
        return

    print("✅ Final Result:", answer)
    update_corrected_text(image_id, answer)

if __name__ == "__main__":
    main()
```

Code Walkthrough

AI Integration:

- Sends LaTeX prompt to Mistral API (local/remote)
- Handles timeouts/errors with fallback model

Response Parsing:

- Extracts solution from different response formats

DB Update:

- Stores AI-generated solutions

App.py

Core pipeline of mathAI

Handles image uploads → OCR (LaTeX) → AI solution (Mistral) → DB storage → Cleanup

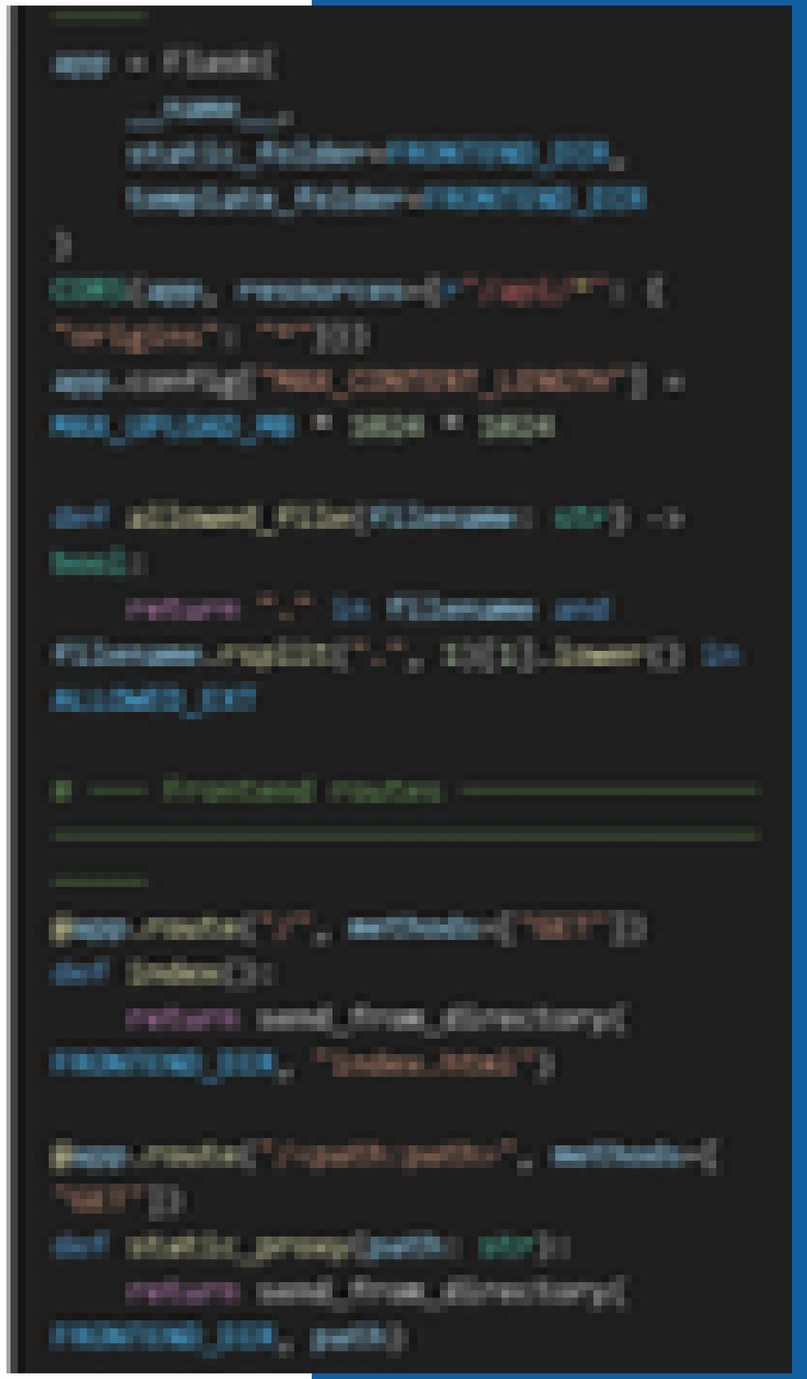
Code Walkthrough:

Setup: Loads env vars, configures Flask/CORS, defines paths.

Routes:

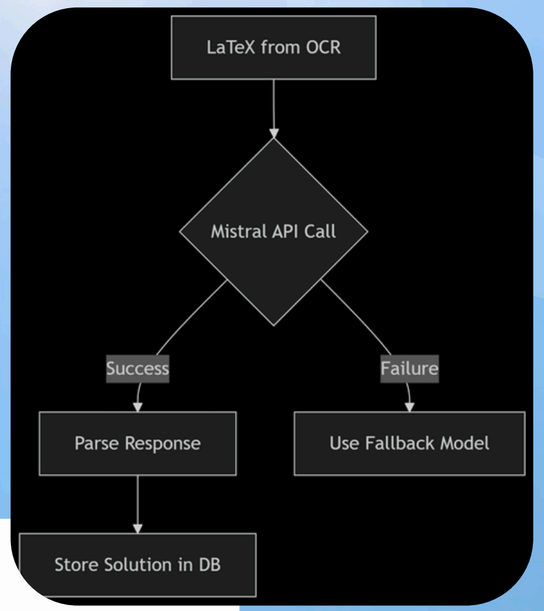
- / → Serves frontend.
- /api/solve → Processes uploads (OCR + AI solver).

Flow:
Validates upload → Saves image → Stores in DB → Extracts math (OCR) → Solves via Mistral → Cleans up → Returns JSON.



Mistral AI Agent

a high-performance, open-source large language model (LLM) designed for efficiency and accuracy in tasks like math, coding, and reasoning.



Architecture

- Open-weight LLM
- Optimized for reasoning & code generation

MathAI Boost

✓ Solves LaTeX math problems accurately
✓ Cost-effective

Key Specs

- Context: 32k tokens
- Quantization: 4-bit/8-bit support
- Speed: ~20 tokens/sec on consumer GPUs

Risks and Challenges

While building the app, we faced several technical and practical challenges, the main challenges are:

Local LLM

Deploying our LLM (Mistral AI) affects the efficiency of solving math problems, compared to the strength of working via API

Integration Issues

Coordinating smooth data flow between frontend, backend, OCR, and AI engine required careful API design and testing

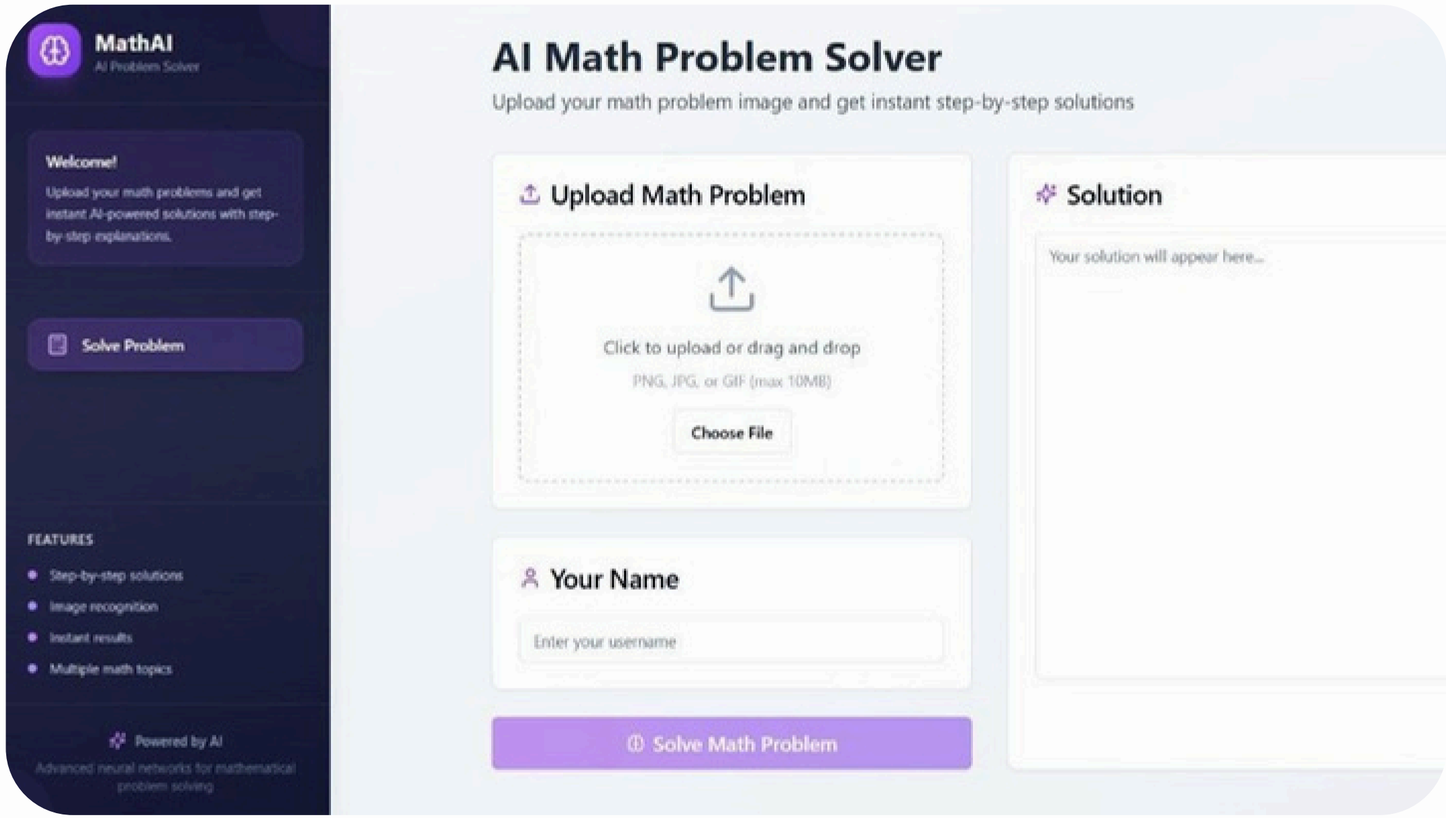
Input Varaibility

Different image qualities, lighting conditions, and angles introduced variability that affected solution accuracy

Processing Time

Delays due to multiple processing steps needed to balance accurate results with fast response. A timeout had been set.

DEMO WALKTHROUGH



MathAI Application Interface

Everything starts easily, just upload the input image containing the math problem

Concept In Business

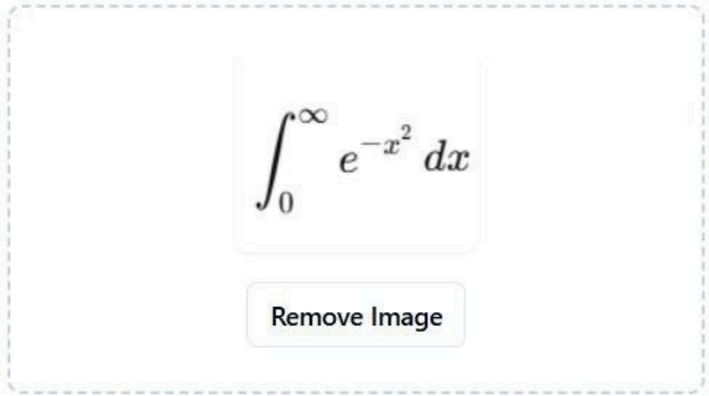
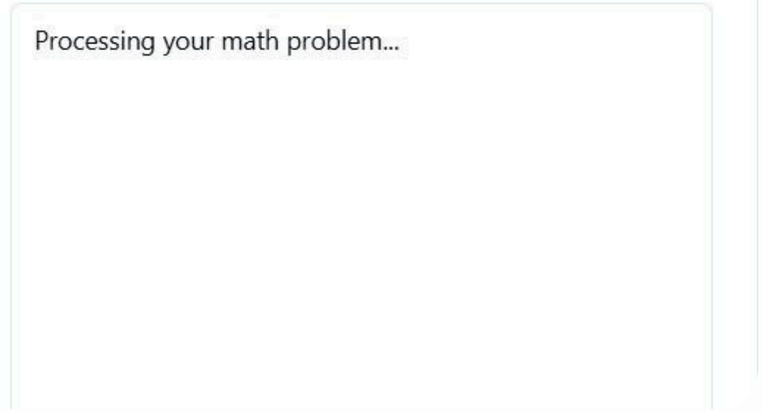


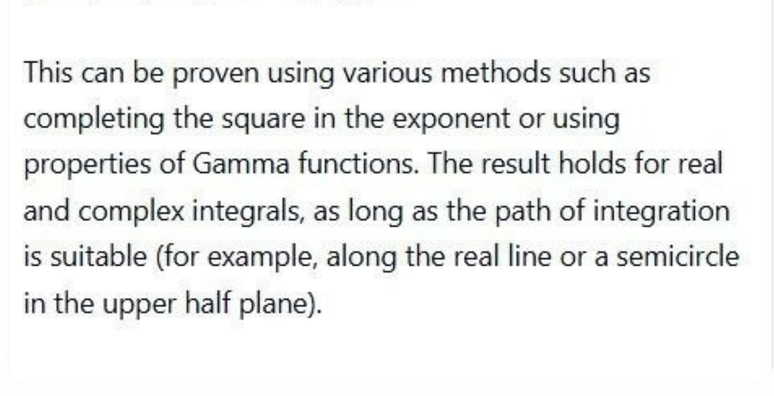
Image Uploaded

This is a complex math problem containing both integral and exponential concepts



Solving Process

It may take from few seconds up to 2 min maximum - depends on complexity of problem



Problem Solved!

Solution provided along with a brief explanation

Future Work

Handwritten Input

Integrate advanced OCR to accurately handle handwritten problems

Strong LLM

Replace the local model with a more powerful cloud-based LLM (via API) for better performance and scalability

Improve AI

Fine-tune prompts and expand support for more complex math topics and formats

Learning Analytics

Track user performance and suggest personalized practice problems.

THANK YOU

ANY QUESTION?

Hasan Chreim 6505
Hussein Rkein 5838