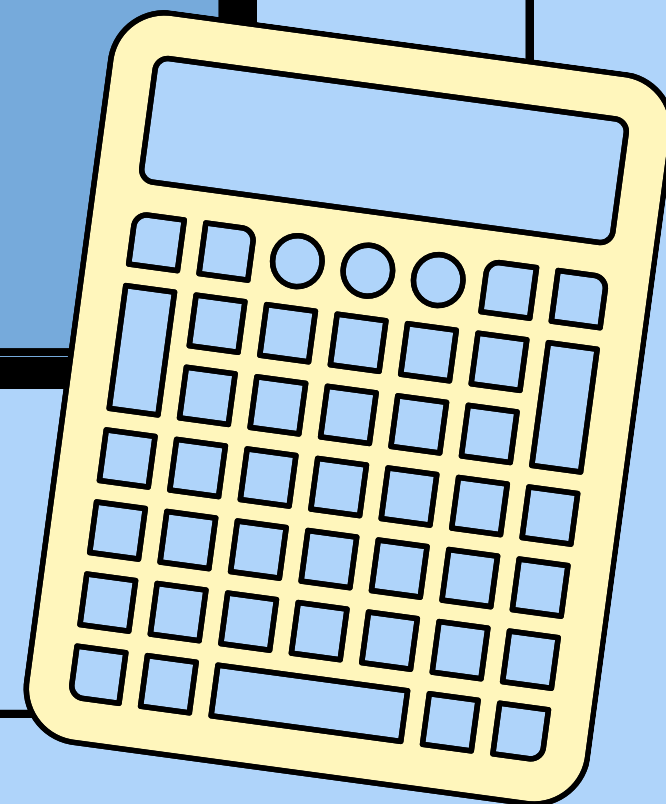


TITIAMATH

Maths - Elementary school
- LLM with IA Gemini 1.5 Flash



SUMÁRIO

1

Introdução

2

Metodologia

3

Resultados

4

Conclusão

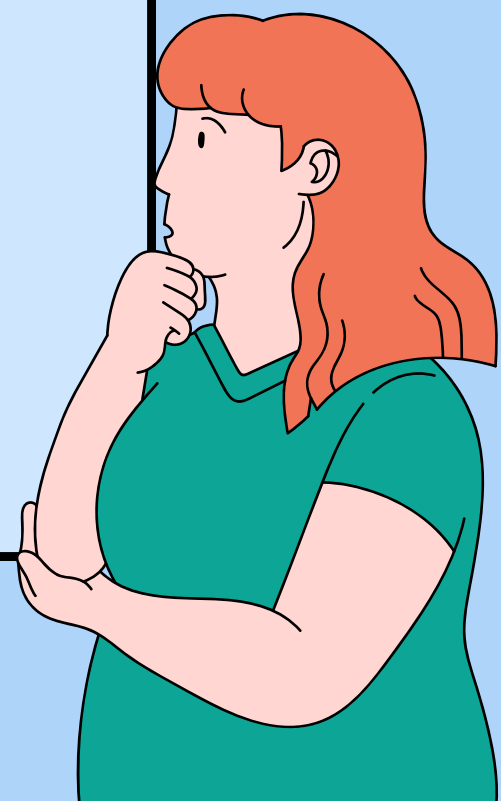


INTRODUÇÃO

No contexto educacional, o uso da IA tem desempenhado um papel crescente e transformador, uma vez que essas tecnologias vêm sendo amplamente aplicadas para personalizar a aprendizagem, aprimorar o processo de ensino e enriquecer a experiência de alunos e professores.

Aplicações de IA LLMs em sistemas de tutoria têm se tornado cada vez mais avançadas e promissoras, especialmente no campo da educação personalizada, sendo empregadas de diversas formas para melhorar a experiência de aprendizado e o suporte ao estudante

Programa Internacional de Avaliação de Estudantes (PISA), segundo os quais apenas 27% dos alunos brasileiros alcançaram o nível 2 de proficiência em matemática, considerado o patamar mínimo de aprendizado.



JUNTANDO IA E MATEMÁTICA

- FOCO EM MELHORAR O DESEMPENHO
- GEMINI
- TELEGRAM
- CHATBOT




METODOLOGIA

- Modelo: LLM
- Chaves de acesso
- Bibliotecas
- Integração



telebot 0.0.5


```
pip install telebot
```



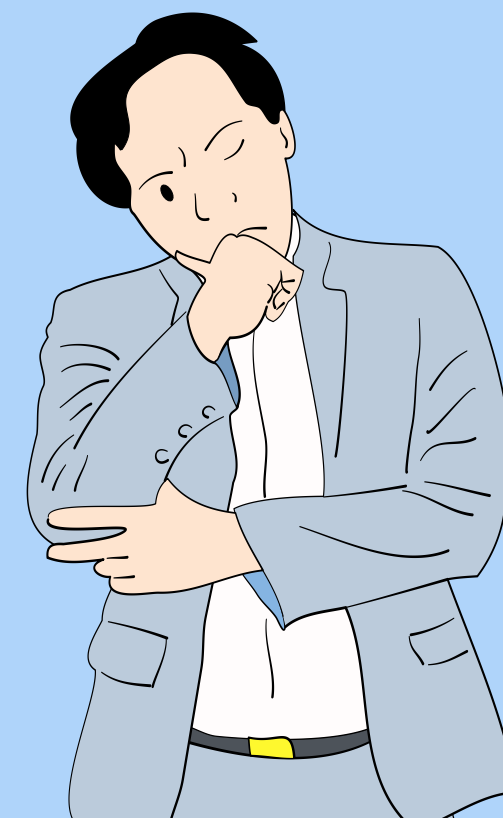
```
pip install telebot
```

google-generativeai 0.8.3

```
pip install google-generativeai
```

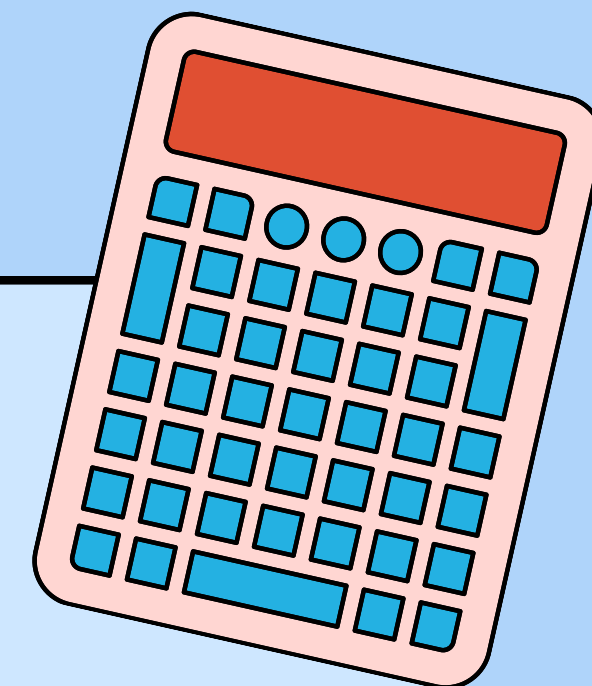


```
pip install google-generativeai
```



METODOLOGIA

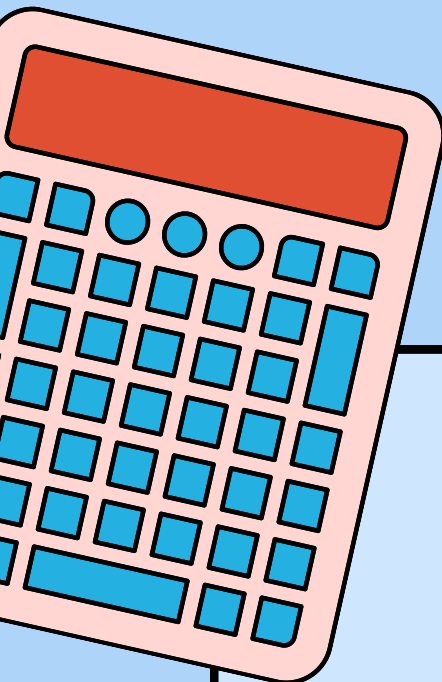
INTEGRAÇÃO



```
1  # Carrega as variáveis de ambiente do arquivo .env
2  load_dotenv()
3
4  # Obtém a variável de ambiente 'APIKEY' para o Google Generative AI
5  api_key = os.getenv('APIKEY')
6
7  # Configura o modelo com a chave da API
8  genai.configure(api_key=api_key)
9  model = genai.GenerativeModel('gemini-1.5-flash')
10
11 # Configuração do bot do Telegram
12 API_KEY_TELEGRAM = os.getenv('TELEGRAMKEY') # A chave de API do Telegram
13
14 bot = telebot.TeleBot(API_KEY_TELEGRAM)
```

METODOLOGIA

FUNCIONAMENTO: GERENCIAMENTO DO HISTORICO



```
1  # Função para salvar o histórico em um arquivo JSON
2  def save_history_to_file():
3      try:
4          with open("user_history.json", "w", encoding='utf-8') as file:
5              json.dump(user_history, file, indent=4, ensure_ascii=False)
6      except Exception as e:
7          print(f"Erro ao salvar o histórico: {e}")
```



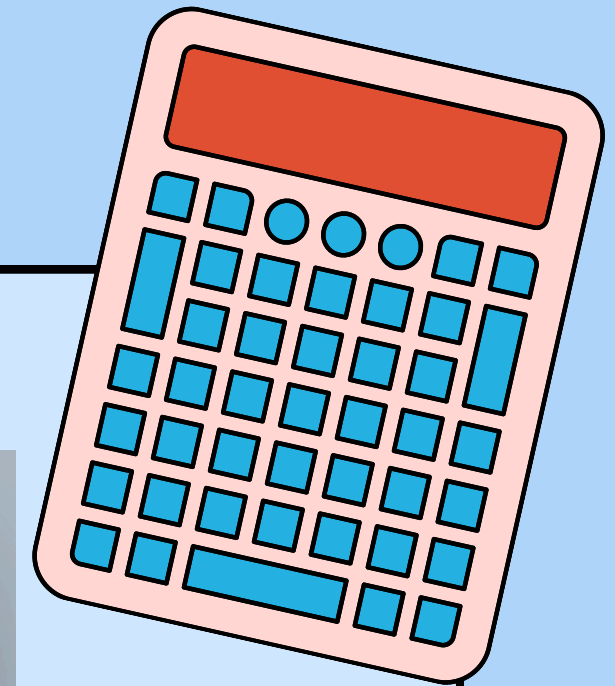
METODOLOGIA

FUNCIONAMENTO: GERENCIAMENTO DO HISTORICO

```
1  # Função para carregar o histórico de um arquivo JSON
2  def load_history_from_file():
3      global user_history
4      try:
5          with open("user_history.json", "r", encoding='utf-8') as file:
6              user_history = json.load(file)
7      except (FileNotFoundError, json.JSONDecodeError):
8          user_history = {}
```


METODOLOGIA

FUNCIONAMENTO: VERIFICAÇÃO DE MATRICULA

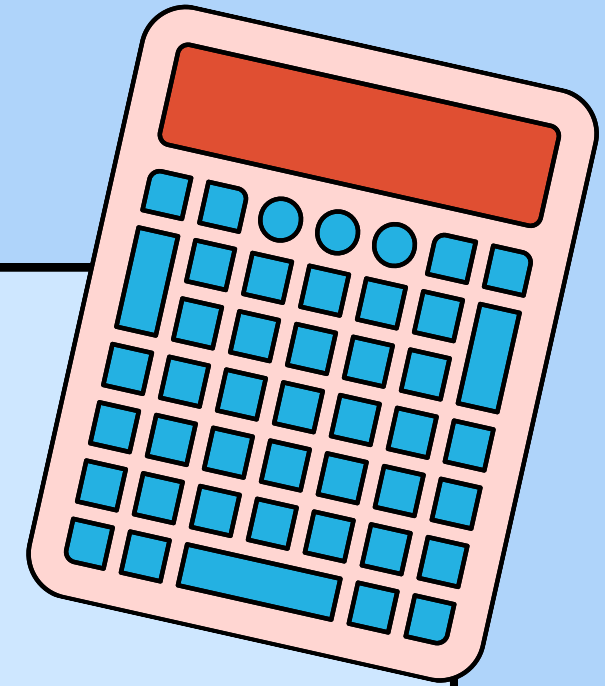


```
1  # Verifica se a matrícula existe no histórico
2  def verify_matricula(message):
3      chat_id = message.chat.id
4      matricula = message.text.strip()
5
6      # 🚫 Verificação do tamanho da matrícula (exatamente 8 dígitos)
7      if not matricula.isdigit() or len(matricula) != 8:
8          msg = bot.reply_to(message, "Matrícula inválida! Ela deve conter exatamente 8 dígitos numéricos. Tente novamente:")
9          bot.register_next_step_handler(msg, verify_matricula) # Solicita novamente
10         return
11
12     if matricula in user_history:
13         user_data = user_history[matricula]
14         bot.reply_to(message, f"Bem-vindo de volta, {user_data['nome']}! Como posso ajudá-lo hoje?")
15
16         # ⚡ Armazena a matrícula corretamente para futuras interações
17         pending_registrations[chat_id] = matricula
18     else:
19         bot.reply_to(message, "Matrícula não encontrada. Vamos fazer seu registro.")
20         pending_registrations[chat_id] = matricula # Armazena a matrícula para continuar o registro
21         msg = bot.reply_to(message, "Por favor, digite seu nome:")
22         bot.register_next_step_handler(msg, process_name)
```

METODOLOGIA

FUNCIONAMENTO: CRIAÇÃO DE MATRICULA

```
1 # Processo de registro de novo usuário
2 def process_name(message):
3     chat_id = message.chat.id
4     nome = message.text.strip()
5     matricula = pending_registrations.get(chat_id)
6
7     if not matricula:
8         bot.reply_to(message, "Erro ao recuperar a matrícula. Tente novamente usando /start.")
9         return
10
11     user_history[matricula] = {"nome": nome, "turma": None, "professor": None, "interacoes": []}
12
13     msg = bot.reply_to(message, f"Obrigado, {nome}! Agora, envie o nome da sua turma:")
14     bot.register_next_step_handler(msg, process_class, matricula)
15
16 def process_class(message, matricula):
17     turma = message.text.strip()
18     user_history[matricula]["turma"] = turma
19
20     msg = bot.reply_to(message, f"Entendido! Agora, envie o nome do(a) professor(a):")
21     bot.register_next_step_handler(msg, process_teacher, matricula)
22
23 def process_teacher(message, matricula):
24     professor = message.text.strip()
25     user_history[matricula]["professor"] = professor
26
27     bot.reply_to(message, f"Registro completo! Bem-vindo(a), {user_history[matricula]['nome']}!")
28
29 # Salva o histórico após o registro
30 save_history_to_file()
```



METODOLOGIA

Restrição da IA:

Constante **MATH_CONTEXT** que especifica ao bot que ele só deve responder a questões sobre matemática fundamental.

```
1  # Contexto restrito para matemática do ensino fundamental
2  MATH_CONTEXT = ""
3  Você é um tutor de matemática especializado no ensino fundamental (1º ao 9º ano).
4  Responda apenas a perguntas relacionadas a matemática desse nível educacional.
5  Se a pergunta não for relacionada a matemática ou estiver fora do escopo, responda:
6  "Desculpe, só posso ajudar com matemática do ensino fundamental."
7  ""
```



METODOLOGIA

Resposta do BOT juntamente da IA:

Para tratar qualquer mensagem enviada ao bot após o login, foi definido um manipulador de mensagens através da função `respond_to_message`. Este manipulador organiza e registra as comunicações com os usuários.

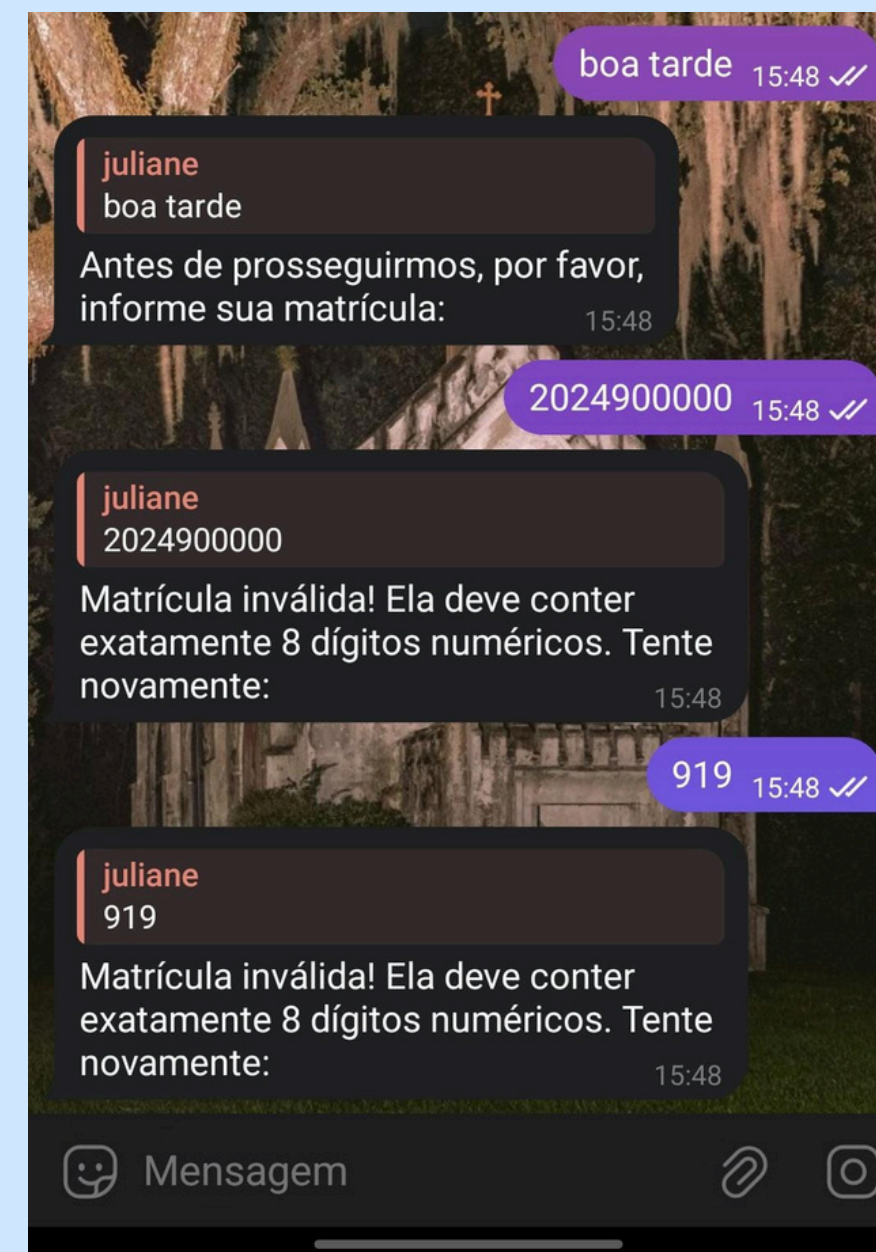
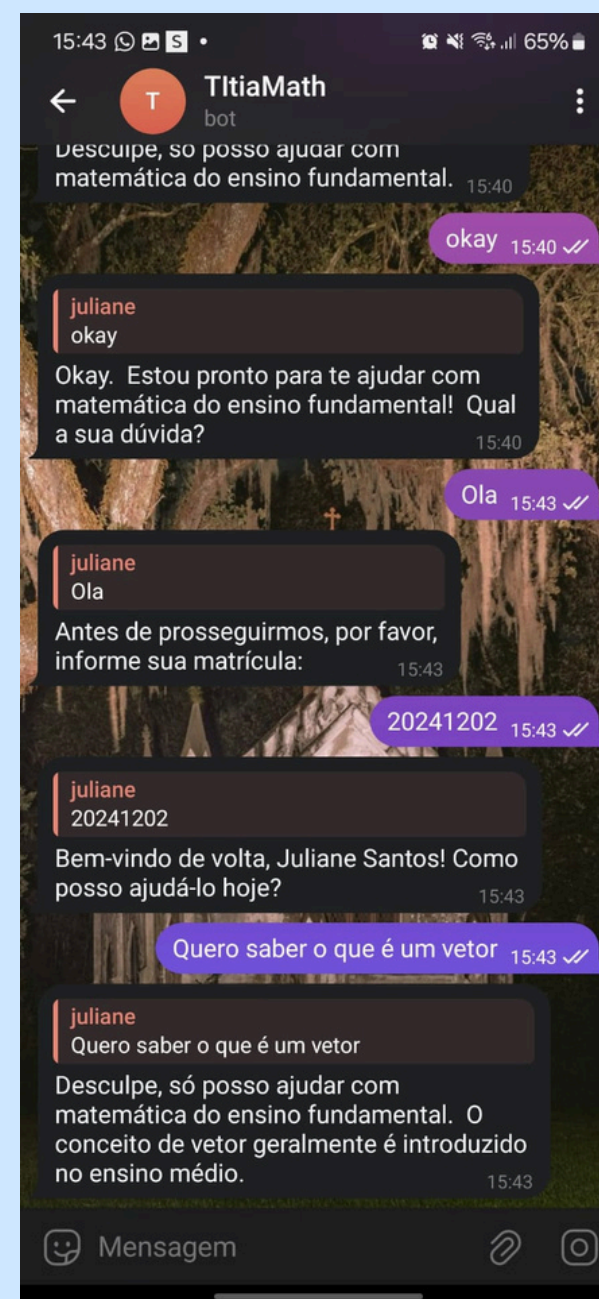
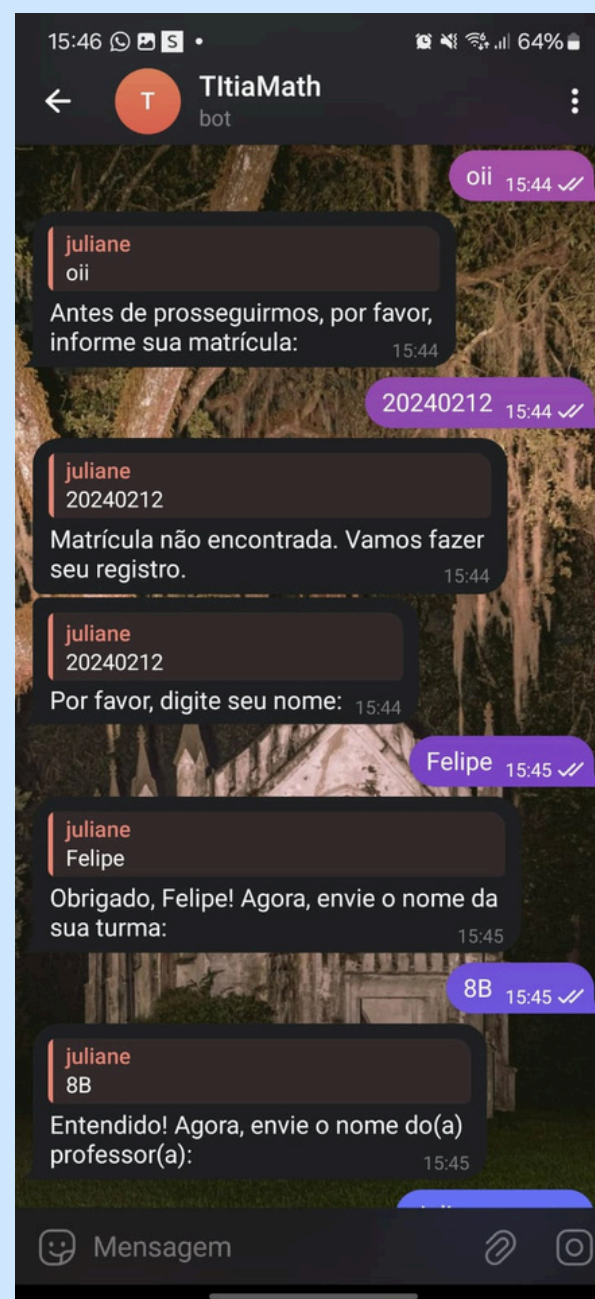
```
1 # Manipulador de mensagens para interação normal após o login
2 @bot.message_handler(func=lambda message: True)
3 def respond_to_message(message):
4     chat_id = message.chat.id
5
6     # Verifica se temos a matrícula salva para esse chat_id
7     if chat_id not in pending_registrations:
8         msg = bot.reply_to(message, "Antes de prosseguirmos, por favor, informe sua matrícula:")
9         bot.register_next_step_handler(msg, verify_matricula)
10        return
11
12    # Recupera a matrícula salva para esse usuário
13    matricula = pending_registrations[chat_id]
14
15    # ⚡ Verifica corretamente se a matrícula existe no histórico
16    if matricula in user_history:
17        user_data = user_history[matricula]
18
19        # Adiciona a interação ao histórico
20        user_data["interacoes"].append({"role": "user", "text": message.text})
21
22        try:
23            # Envia o prompt com contexto para a IA
24            response = model.generate_content(MATH_CONTEXT + "\nUsuário: " + message.text + "\nIA:")
25
26            # Adiciona a resposta ao histórico
27            user_data["interacoes"].append({"role": "bot", "text": response.text})
28
29            # Envia a resposta ao usuário
30            bot.reply_to(message, response.text)
31
32        except Exception as e:
33            bot.reply_to(message, "Houve um problema ao processar sua solicitação. Tente novamente mais tarde.")
34
35        # Salva o histórico após a interação
36        save_history_to_file()
37    else:
38        bot.reply_to(message, "Matrícula não reconhecida. Por favor, inicie novamente com /start.")
```

RESULTADOS

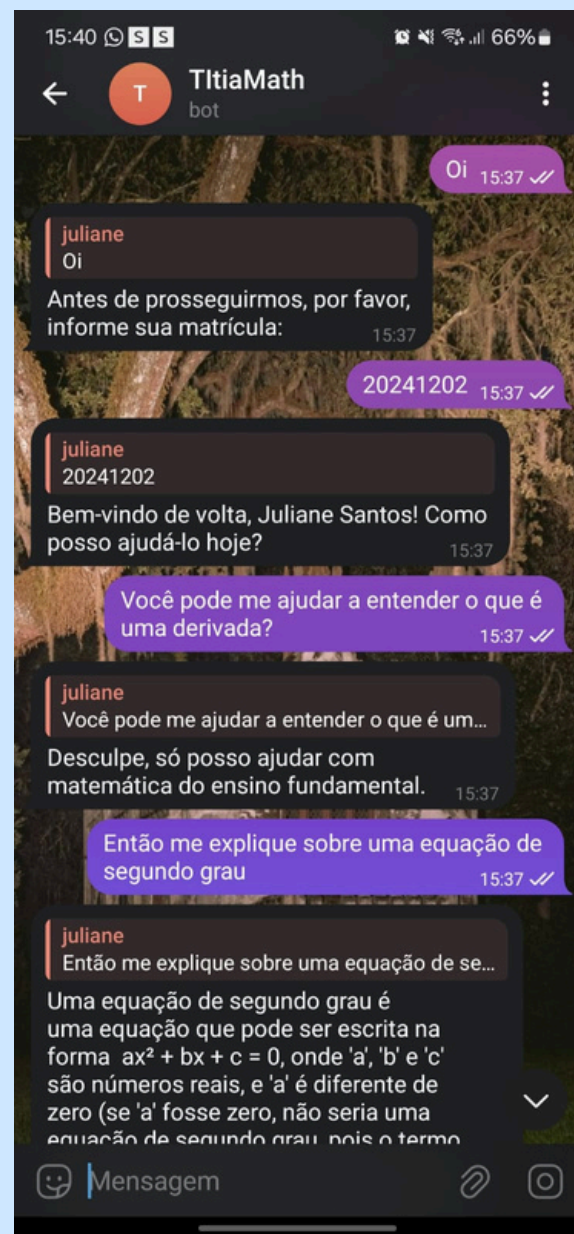
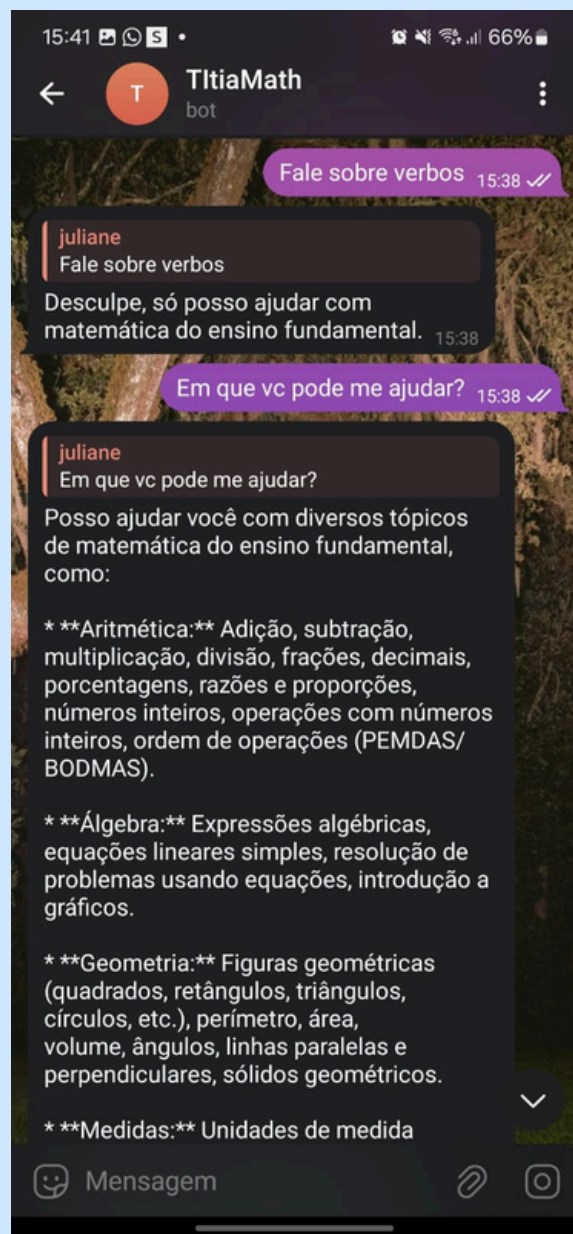
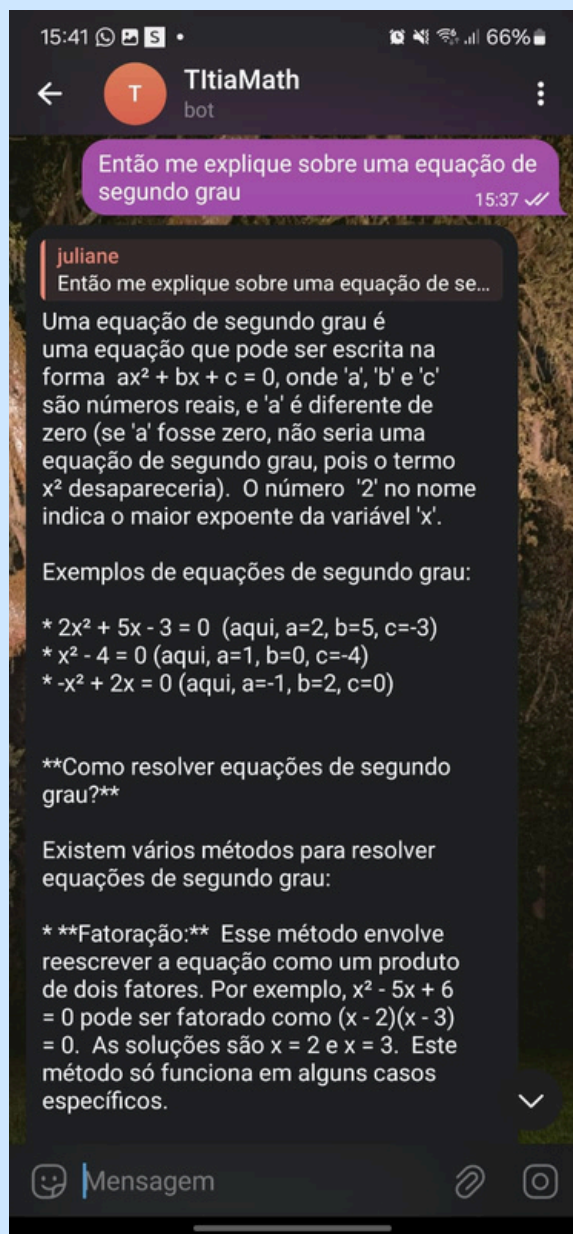
- Considerando que o modelo utilizado já é pré-treinado, não foi necessário realizar um novo treinamento ou aplicar métricas tradicionais de avaliação.
- Foram realizados alguns testes com o chatbot para verificar a acurácia e assertividade das respostas geradas, bem como avaliar as restrições implementadas para garantir que ele forneça exclusivamente conteúdos de matemática voltados ao ensino fundamental.



RESULTADOS



RESULTADOS



CONCLUSÃO

- O uso da IA na aprendizagem oferecer aos seus usuários uma vasta rede de conteúdos de forma rápida e eficaz.
- Os Sistemas de tutoria inteligente, ampliam ainda mais as possibilidades de melhoria no aprendizado.
- A Aplicação é funcional e acessível aos alunos, permitindo que os alunos esclareçam suas dúvidas de forma eficiente, contribuindo significativamente para o aprimoramento do aprendizado na disciplina de matemática.

REFERÊNCIAS

AGÊNCIA BRASIL. Menos de 50% dos alunos sabem o básico em matemática e ciências. Disponível em: <https://agenciabrasil.ebc.com.br/educacao/noticia/2023-12/menos-de%2050%25-dos-alunos-sabem-o-b%C3%A1sico-em-matem%C3%A1tica-e-ci%C3%Aancias>. Acesso em: 15 jan. 2025.

BARROSO, Luís Roberto; MELLO, Patrícia Perrone Campos. Inteligência artificial: promessas, riscos e regulação. algo de novo debaixo do sol. Revista Direito e Práxis, [S.L.], v. 15, n. 4, p. 1-45, jun. 2024. FapUNIFESP (SciELO). <http://dx.doi.org/10.1590/2179-8966/2024/84479>.

RODRIGUES, Olira Saraiva; RODRIGUES, Karoline Santos. A inteligência artificial na educação: os desafios do chatgpt. Texto Livre, [S.L.], v. 16, p. 1-12, jul. 2023. FapUNIFESP (SciELO). <http://dx.doi.org/10.1590/1983-3652.2023.45997>.

SANTAELLA, Lucia. A inteligência artificial é inteligente? São Paulo: Almedina, 2023.