Al in Parenting: The GPT-Powered Parenting Assistant

#ParentingAssistant #GPT #AIPoweredParenting #AlinChildcare #SmartParenting #ParentingTechnology #ChildcareInnovation #ParentingSupport #AlinParenting #NextGenParenting #OpenAIParenting #ParentingRevolution #DigitalParenting #AIForBetterParenting #ParentingMadeEasier #Multiligualassistant

Author: VANI KRISHNAN

Github: https://github.com/VANIRAMAKRISHNAN/parentingassistant.git

Problem Description:

Parenting, particularly for first-time parents, can be a daunting task. Parents often find themselves overwhelmed by the myriad of responsibilities and decisions that they have to make on behalf of their newborn. The task becomes even more complex when parents have to figure out the reasons behind their child's behavior, like frequent crying, or tracking their child's growth and development. Adding to the challenge, some parents might not be fluent in English, the dominant language in most parenting resources. This brings us to the need for a **multilingual parenting assistant** that can provide **on-demand**, **accurate** and **helpful information**, **especially during off-hours**.

Inspiration:

The primary inspiration for this project was the mothers of newborns who require immediate assistance during off-hours. They are often left clueless when their baby starts crying at odd hours or when they need to track the child's progress but don't have the means to do so efficiently. This app, in regional languages, can provide them with the assistance they need.

Solution:

Our solution is a multilingual 'Parenting Assistant' web app powered by GPT and Translator API. This app will be designed to understand questions about parenting in any regional language and provide responses in the same language. These responses will be curated by leveraging the power of AI - particularly OpenAI's GPT model - which is known for generating human-like text based on the input provided.

Architecture:

Parenting Assistant



Pre-requisites:

1. We'll set up Translator API for the language conversion. Here, we will be using the Microsoft Azure Translator Text API.

To use the Microsoft Azure Translator Text API, you'll need to set up an Azure account, create a new resource for the Translator Text API, and grab the API key. Create Translator API in Azure as shown below:

← → C 🔒 portal.azure.com/#create/Microsoft.CognitiveServicesTextTranslation							
<u>Microsoft Azure</u>			1				
Home > Cognitive Services Translator	>						
Create Translator							
Basics Network Identity Tags	Review + create						
Easily integrate real-time text translation capabilities into your application's websites, tools, or any solution requiring multi-language support such as website localization, e-commerce, customer support, messaging applications, internal communication, and more.							
Project Details							
Subscription * ①	Visual Studio Enterprise Subscription	\checkmark					
Resource group * ()	parenting_assistant	\checkmark					
Instance Details	Creste new						
Plesse choose the Global region unless your business or application requires a specific region. Applications that do not offer a region solection use the Global region.							
Region * ()	East US	×					
Name * ①	parentingassistant	~					
Pricing tier *	Free F0 (Up to 2M characters translated per month)	$\overline{}$					
View full pricing details							

Capture the Keys, region, endpoint of the Translator API

nt	rosoft Azure	Search resources, services, and	docs (G+/)
 Delete C Cance + Redeploy 2 Download (Refresh Your deployment is complete Deployment name : Microsoft CognitiveServicesTextTranslation-20230528083316 Subscription : <u>Visual Studio Enterprise Subscription</u> Resource group : <u>parenting assistant</u> Deployment details Next steps 	crosoft.CognitiveS	ServicesTextTranslation-20230528083316 Overview 🖈	
		Vour deployment is complete Peployment name : Microsoft.CognitiveServicesTextTranslation-20230528083316 Subscription :: Visual Studie Enterprise Subscription Resource group : parenting_assistant Deployment details Next steps	Start time : 5/28/2023, 8:34:51 AM Correlation ID : fc8a352a-a308-4b56-9b60-3606449c4c4b

2. Ensure that you have your OpenAI API key at hand, which is necessary for making API requests. Generate OpenAI - API key from OpenAI website.

Implementation Detail:

As part of this implementation, first, lets setup in Azure VM (Linux - Ubuntu). Post this, we can containerise and deploy to AKS

Step 1: Azure VM setup:

- 1. Create Resource Group
- 2. Create a Azure VM (Linux Ubuntu VM)
- 3. Open port 5000 in network to allow traffic
- 4. Install Python libraries by executing the below commands in sequence:

sudo apt update sudo apt install python3-pip pip3 install Flask openai

Step 2: Setting Up the Backend

We will start with setting up the Flask backend, Python being the server-side language. We'll create a POST API endpoint '/ask' to accept the user's parenting question in regional language, translate it to English, feed it to GPT, and then translate the response back to the original language.

Create a new file named '**app.py**' with below code snippet:

```
import os
import openai
import requests
import uuid
import json
from flask import Flask, render_template, request, jsonify, send_file
app = Flask(__name__)
openai.api_key = "provide openai api key"
# Set up Azure translator
subscription_key = 'provide translator key'
endpoint = 'https://api.cognitive.microsofttranslator.com/'
location = 'provide region'
def translate(input_text, target_language):
  path = '/translate'
  constructed_url = endpoint + path
  params = {
    'api-version': '3.0',
    'from': 'en',
    'to': [target_language]
  }
  headers = \{
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Ocp-Apim-Subscription-Region': location,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
  }
  body = [{'text': input_text}]
  request = requests.post(constructed_url, params=params, headers=headers, json=body)
  response = request.json()
  return response[0]['translations'][0]['text']
@app.route('/')
def index():
  return render_template('index.html')
@app.route('/ask', methods=['POST'])
def ask():
  data = request.get_json() # Use get_json() since we're sending JSON
  question = data['question']
  language = data['language']
```

Translate question to English
question_in_english = translate(question, 'en')

```
response = openai.Completion.create(
    engine="text-davinci-002",
    prompt=f"Parenting question: {question_in_english}\nAnswer:",
    max_tokens=150,
    n=1,
    stop=None,
    temperature=0.5,
)
```

```
answer = response.choices[0].text.strip()
```

```
# Translate answer to the chosen language
answer_in_chosen_language = translate(answer, language)
```

```
return jsonify({'answer': answer_in_chosen_language})
```

```
if __name__ == '__main__':
    app.run(debug=True, host="0.0.0.0")
```

Step 3: Setting Up the Frontend

We will create a basic HTML page (**index.html**) under a folder "**templates**" with a form to input the user's question and a space to display the assistant's response. CSS will be used for styling, and JavaScript to make the API call.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Parenting Assistant - powered by GPT</title>
  k href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap"
rel="stylesheet">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <style>
    body {
      font-family: 'Roboto', sans-serif;
      background-color: #f06292;
      color: white;
      display: flex;
      align-items: center;
      justify-content: center;
      height: 100vh;
      margin: 0;
```

```
padding: 0;
    }
    #container {
      text-align: center;
      padding: 20px;
      background: rgba(0,0,0,0.7);
      border-radius: 10px;
    }
    h1 {
      margin-bottom: 20px;
    }
    #chatbox {
      height: 300px;
      width: 100%;
      overflow-y: scroll;
      margin-bottom: 20px;
      border: 1px solid white;
      padding: 10px;
      background: white;
      color: black;
    }
    input, button, select {
      padding: 10px;
      font-size: 18px;
    }
    input {
      width: 60%;
    }
    select {
      width: 20%;
    }
    button {
      width: 18%;
      margin-left: 2%;
      background: #ff4081;
      color: black;
   }
  </style>
</head>
<body>
  <div id="container">
    <h1>Parenting Assistant - know about your child</h1>
    >अपने बच्चे के बेहतर भविष्य के लिए समर्पित
    <div id="chatbox">
    </div>
    <form id="chat-form">
      <select id="language" required>
        <option value="" disabled selected>Select your language</option>
```

```
<option value="en">English</option>
        <option value="hi">Hindi</option>
        <option value="ta">Tamil</option>
        <option value="ml">Malayalam</option>
        <option value="te">Telugu</option>
        <option value="kn">Kannada</option>
      </select>
      <input type="text" id="question" placeholder="Type your question here..." required>
      <button type="submit">Ask</button>
    </form>
 </div>
 <script>
    $(document).ready(function() {
    $('#chat-form').on('submit', function(e) {
      e.preventDefault(); // Prevent the form from being submitted normally
      let question = $('#question').val(); // Get the value of the question input
      // Send a POST request to the server
      $.ajax({
        url: '/ask',
        method: 'POST',
        contentType: 'application/json',
        data: JSON.stringify({
          question: question,
          language: $('#language').val() // Include the selected language
        }),
        dataType: 'json',
        success: function(data) {
          let answer = data.answer; // Get the answer from the response
          // Add the question and answer to the chatbox
          $('#chatbox').append('You: ' + question + '');
          $('#chatbox').append('Al: ' + answer + '');
          // Clear the question input
          $('#question').val('');
        },
        error: function() {
          // This function will be called if the request fails
          alert('An error occurred. Please try again.');
        }
      });
   });
 });
 </script>
</body>
</html>
```

Step 4: Execute the program app.py like below:

python3 app.py

Application will be accessible as shown below:

User can select the language of their choice and ask questions in regional language as shown below. Respective responses from GPT will be reverted back.

Translator API enables the language conversion from regional to English and vice versa

← → C ▲ Not Secure 20.83.165.88:5000				ů \$	@ 1) = I	🛛 🖪 Update	
Ρ	arenting Ase	sistant - know abo	ut your child					
अपने बच्चे के बेहतर भविष्य के लिए समर्पित								
	You: What is th	he normal weight of a 3 year old girl l	based in India					
Al: There is no one "normal" weight for a 3 year old girl in India, as children can vary widely in size and weight depending on their individual genetics and growth patterns. However, according to the World Health Organization's growth standards, the average weight for a 3 year old girl in India is about 13.5 kg.								
You: എന്റെ കുഞ്ഞിന് വളരെ കൂടുതൽ സ്വപ്നം കാണുന്നു. ഇത് സാധാരണമാണോ?								
AI: ഈ ചോദ്യത്തിന് ഒരു വലുപ്പത്തിലുള്ള ഉത്തരമില്ല, കാരണം ഓരോ കുട്ടിയും വ്യത്യസ്തമാണ്, അതിനാൽ വ്യത്യസ്ത ഉറക്ക ആവശ്യങ്ങൾ ഉണ്ടാകും. എന്നിരുന്നാലും, നിങ്ങളുടെ കുട്ടി അതേ പ്രായത്തിലുള്ള മറ്റ് കുട്ടികളേക്കാൾ കൂടുതൽ ഉറങ്ങുകയാണെങ്കിൽ, ഏതെങ്കിലും അടിസ്ഥാന ആരോഗ്യ പ്രശ്നങ്ങൾ തള്ളിക്കളയാൻ നിങ്ങളുടെ ശിശുരോഗവിദദ്ധനുമായി ചർച്ച ചെയ്യുന്നത് മൂല്യവത്താണ്.								
You: నా పిల్లకు చాలా ఎక్కువ కలలు వస్తున్నాయి. ఇది సాధారణమేనా?								
Al: పిల్లలు చాలా ఎక్కువ కలలు వస్తున్నాయే సాధారణం కాదు. మీ ప								
		0 – u						
Telugu	Type your question	here						
		Ask						

Services used for this blog:

← → C Portal.azure.com/#view/HubsExtension/RecentResources.ReactView					⊵	Q	Ů	☆	(F)	*	≡ſ	
Microsoft Azure	P Search resources, services, and docs (G+/)							۶.	a 6	D ©	0	R
Home >												
Recent 🖈 …												
O Refresh 🔞 Clear 📈 Freedback 🙆 Assign tags												
V Filter for any field.												
Showing 7 results.												
Name Type		Location	Resource Group	Subscription					Last acc	essed		
🔲 🖳 parentingassistantvm … Virtual	l machine	East US	parenting assistant	Visual Studio	Enterpris	e Subscri	ption		25 minu	ites ago		
parentingassistantazure ···· Transla	ator	Southeast Asia	parenting_assistant	Visual Studio	Enterpris	e Subscri	otion		26 minu	ites ago		
parentingassistanttranslator ···· Translat	ator	East US	parenting_assistant	Visual Studio	Enterpris	e Subscri	otion		11 hour	s ago		
[iii] <u>parenting_assistant</u> ··· Resour	rce group		parenting_assistant	Visual Studio	Enterpris	e Subscri	ption		11 hour	s ago		
📄 🕈 Visual Studio Enterprise Subscription Subscription	ription			Visual Studio	Enterpris	e Subscri	ption		2 days a	igo		
DefaultResourceGroup_SEA ··· ResourceGroup_SEA	rce group	Southeast Asia	DefaultResourceGroup-SEA	Visual Studio	Enterpris	e Subscri	ption		14 days	ago		
(ii) <u>NetworkWatcherRG</u> ···· Resour	rce group	Southeast Asia	NetworkWatcherRG	Visual Studio	Enterpris	e Subscri	ption		14 days	ago		

Using the requirements.txt, Dockerfile and Kubernetes manifest files and ACR, we have deployed the application to AKS for scalability as well.

requirements.txt

flask openai requests

Dockerfile

FROM python:3.7 WORKDIR /app COPY requirements.txt /app RUN pip install -r requirements.txt COPY . /app EXPOSE 5000 CMD ["python", "app.py"] **Kubernetes manifest files:**

Deployment.yaml

apiVersion: apps/v1 kind: Deployment metadata: name: parenting-assistant spec: replicas: 3 selector: matchLabels: app: parenting-assistant template: metadata: labels: app: parenting-assistant spec: containers: - name: parenting-assistant image: <your-dockerhub-username>/parenting-assistant:latest ports: - containerPort: 5000

Service.yaml

apiVersion: v1 kind: Service metadata: name: parenting-assistant-service spec: selector: app: parenting-assistant ports: - protocol: TCP port: 80 targetPort: 5000 type: LoadBalancer

Challenges Faced:

We encountered a few challenges during the implementation process.

- **1. Language conversion**: Identifying the solution for language conversion and integrating the same for supporting regional language was a challenge. The efficient translation of questions and answers between regional language and English was a significant challenge, given the complexity of the languages.
- **2. Understanding parenting queries**: Providing relevant answers to parenting questions was another hurdle.
- **3. Ensuring the reliability of the answers**: As this application is about parenting, it is crucial to ensure that the information provided is accurate and safe. Thus, there was a challenge to make sure the AI does not provide misleading information. To combat this, thorough testing and monitoring of the model's responses were conducted.

Benefits:

- **1. 24/7 Assistance**: The parenting assistant can provide constant help to parents, particularly helpful during off-hours when immediate assistance is not available.
- 2. **Multi-language Support**: With the Translator API, the app can cater to non-English speaking parents and can be easily expanded to other languages in the future.
- **3. Personalized Responses**: With the power of GPT, the app provides personalized responses based on the question context, making it more relatable and useful.

Conclusion:

The "Parenting Assistant" web application brings the power of AI and multilingual support to parenting, providing a valuable resource for new parents. Despite the challenges encountered during the development, the final product offers a readily available and easily accessible source of parenting assistance. The blend of AI and human touch in this application creates a valuable and unique resource in the world of parenting.