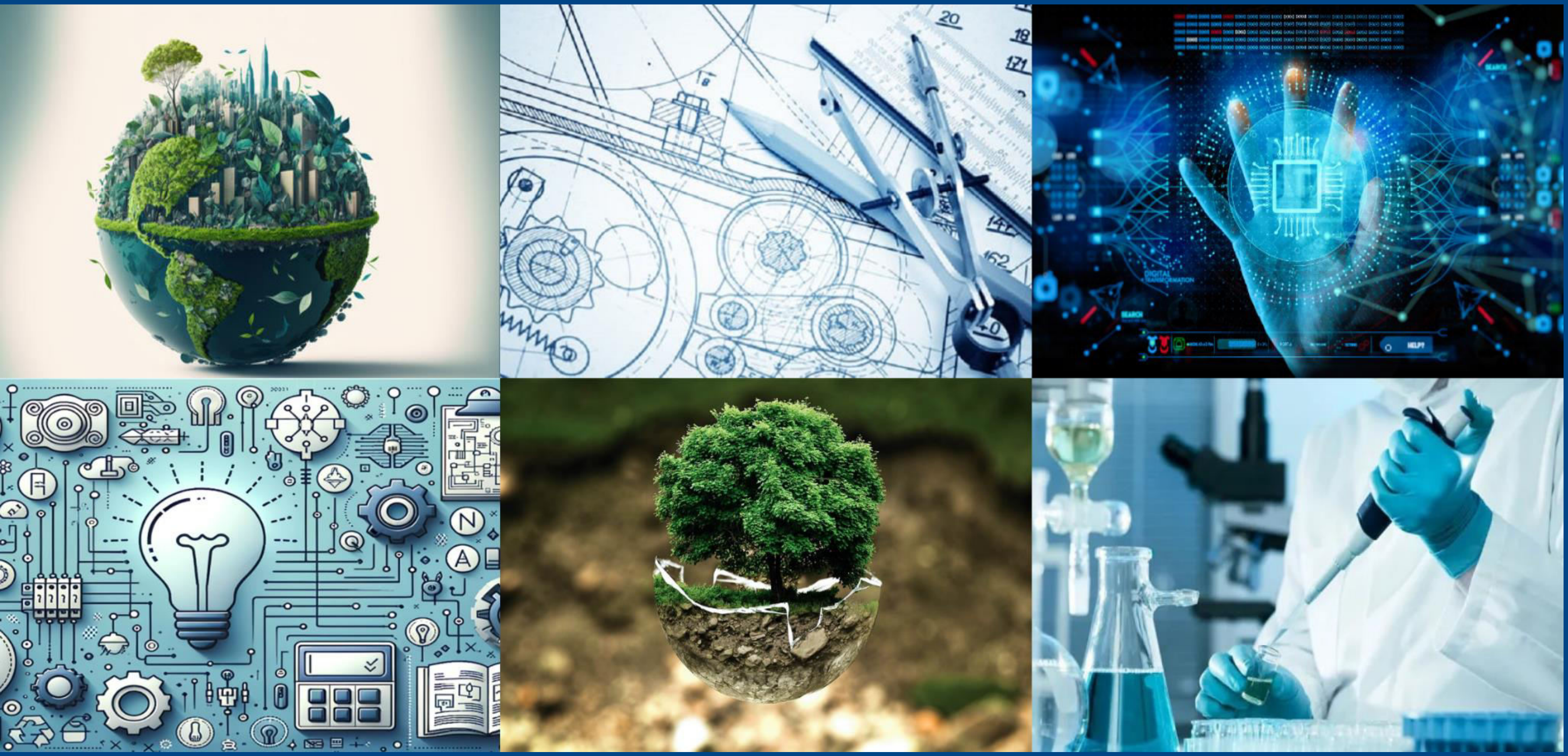
ISSN: 2582-7219

International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.206

Volume 8, Issue 3, March 2025

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

ARV-GPT: AI Virtual Assistant for Heritage Exploration

Mrs.A.Sangeetha, Mrs.N.Sarmiladevi, G.David William, K.Pon Sridharan, T.Sutharsan

Assistant Professor, Dept. of AI&DS, Kongunadu College of Engineering and Technology, Trichy, Tamil Nadu, India

Assistant Professor, Dept. of AI&DS, Kongunadu College of Engineering and Technology, Trichy, Tamil Nadu, India

UG Student, Dept. of AI&DS, Kongunadu College of Engineering and Technology, Trichy, Tamil Nadu, India

UG Student, Dept. of AI&DS, Kongunadu College of Engineering and Technology, Trichy, Tamil Nadu, India

UG Student, Dept. of AI&DS, Kongunadu College of Engineering and Technology, Trichy, Tamil Nadu, India

ABSTRACT: ARV-GPT is a full-stack AI chatbot that is capable of offering smooth and intelligent conversations in English and Tamil. The backend is developed with Django Rest Framework (DRF) which manages user authentication and API integrations. The chatbot uses the Groq API with the LLaMA LLM model to produce responses, providing sophisticated language understanding and contextual precision. The frontend is coded with Flet, a UI framework based on Python that provides cross-platform dynamic and interactive chat application. The application has real-time text-to-speech (TTS) through flet-audio, which enables users to hear AI-driven responses in the language of their choice. Rather than retrieving text response from the server, the TTS component produces speech on the local system, providing for quick and seamless playback. To provide better user experience, a toggle switch is provided to switch between English and Tamil and the translations are taken care of prior to audio playback. Markdown formatting is also removed from responses to facilitate smooth speech synthesis. SQLite, a lightweight and efficient database, powers the entire system to handle user interaction and chat history. This project incorporates async/await processing to manage AI responses, in-real-time audio generation and UI updates without lag. The audio button toggles dynamically to a loading state while processing TTS and a play button upon completion of audio. ARV-GPT is designed for both mobile and web platforms**.**

KEYWORDS: Groq API, Large Language Model (LLM), Llama 3. 2, Django Rest Framework (DRF) API, Django Backend, Flet, flet-audio, TTS and Generative AI.

I. INTRODUCTION

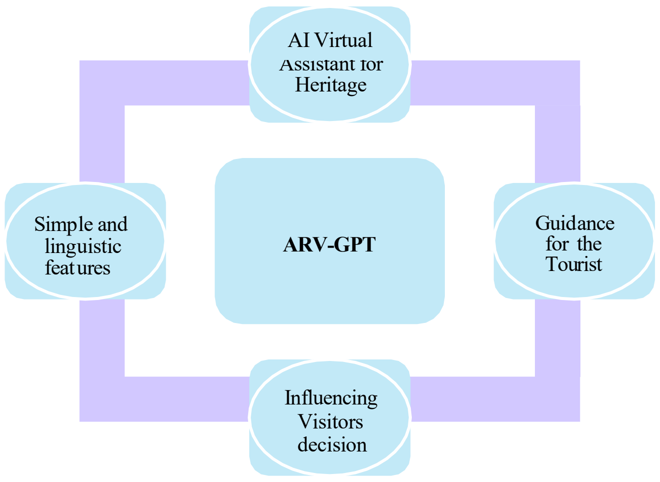
ARV-GPT is a virtual assistant powered by AI that assists users in exploring the extensive heritage places of Tamil Nadu through accurate, real-time and multilingual information. The platform uses the latest AI technologies, such as the Groq API and the LLaMA LLM model, to present informative responses regarding historical monuments, cultural importance, architectural aspects, accessibility and visitor amenities. In contrast to the conventional guidebook or signboard, which tend to offer brief information, ARV-GPT provides a dynamic and customized user experience through AI-driven interaction. The application has a sleek chat interface developed using Flet, where users can input questions and get instant answers in both English and Tamil. The chat interface is made to be easy to use and responsive, with support for natural language processing (NLP) for context-sensitive answers.

Users can input specific questions regarding a heritage site, like history, best times to visit, or attractions nearby and get elaborate, well-formatted answers in their choice of language. ARV-GPT is enhanced with bilingual answers in English and Tamil for improved accessibility. The system has a language toggle button within the UI through which users can switch languages quickly. Furthermore, the system implements real-time text-to-speech (TTS) conversion through flet-audio so that users can listen to the response rather than read it. The feature proves valuable for tourists, visually impaired people, or learners who are fond of learning by listening. To provide clean and natural audio playback, the system removes markdown formatting from AI responses prior to speech conversion.

Users can play the generated audio by clicking on a play button, which briefly changes to a loading animation while the speech file is asynchronously generated. After the audio is generated, the button returns to its original play icon for convenient access. The ARV-GPT backend has been constructed upon Django Rest Framework (DRF) and SQLite, allowing the secure user login, history maintenance of conversations and communication of APIs.

Concurrency support, scalability and robustness towards real-world operation are implemented such that user requests are effectively addressed in real time. Users may access login, register and token-based authentication features in DRF. The queries are routed to the Groq API and the LLaMA LLM model processes and delivers contextually appropriate responses. Audio Processing on the Client Side, Rather than retrieving pre-generated audio files from the server, the Flet UI generates audio dynamically on the client device through flet-audio, saving latency and server load.

In contrast to static information sources, ARV-GPT leverages AI-driven automation to generate answers dynamically in response to user input. This results in a more interactive and immersive system, offering users personalized insights. For instance, users can query "What is the best time to visit Brihadeeswarar Temple?" or "Tell me about the history of Mahabalipuram," and get rich, structured answers instead of boilerplate information. The app is also made scalable with integration facilities of external APIs and databases to continually enhance its knowledge base. Augmented Reality (AR) Integration, Enabling users to visualize 3D virtual reconstructions of heritage locations for better experience.


AI Virtual Assistant for Heritage
Simple and linguistic features
ARV-GPT
Guidance for the Tourist
Influencing Visitors decision

F IGURE 1.1 INTRODUCTION OF ARV-GPT

IOS Compatibility Extending the platform to mobile apps with Flet, providing smooth access on phones and tablets. Applying analytics to monitor user interaction, monitor top heritage searches and personalize responses. The chatbot can be upgraded to display relevant images of historical and heritage sites in Tamil Nadu alongside the text responses. By integrating an image database or fetching images from an external API, users can get a more immersive and visually informative experience when exploring heritage sites. This would help in providing a better understanding of the site's architecture, history and significance.

Integrating Google Maps or OpenStreetMap with the chatbot will allow users to locate heritage sites in real time. The chatbot can provide navigation assistance, including directions, estimated travel time and nearby tourist attractions. This feature will be highly beneficial for tourists who want to explore Tamil Nadu’s cultural sites efficiently. Users will be able to interact with maps, view satellite imagery and access geolocation-based recommendations. Through the use of cutting-edge AI and multilingual capabilities, ARV-GPT seeks to transform heritage tourism, making it more accessible, informative and interactive. Synching real-time AI support, voice interaction and automation, it closes the gap between technology and cultural heritage, providing a contemporary way of accessing Tamil Nadu's heritage.

II. LITERATURE SURVEY

Heritage exploration has traditionally relied on physical guides, printed brochures and static signboards, limiting the accessibility and engagement of historical information. Visitors often face challenges in obtaining real-time, interactive and multilingual information about heritage sites. With advancements in artificial intelligence (AI) and natural language processing (NLP), AI-powered virtual assistants such as ARV-GPT have emerged as innovative solutions for enhancing heritage tourism.

Unlike traditional methods, ARV-GPT ensures that visitors receive interactive, context-aware and historically rich narratives in their preferred language, thereby making heritage knowledge more engaging and accessible to a global audience.

Flet is a Python framework for developing real-time interactive applications with ease and cross-platform compatibility. It allows developers to build web, desktop and mobile apps without frontend knowledge. Flet has a declarative UI like Flutter and is simple to use for building interfaces using Python. It facilitates cross-platform deployment on Windows, macOS, Linux android, iOS and the Web for smooth user experiences. Flet has pre-built widgets like buttons, text fields, containers, rows and columns, which enable developers to rapidly create applications. The framework is capable of real-time UI updates without page reloads, making it suitable for chat apps and interactive interfaces. Event handling in Flet is easy with pre-built methods for handling user input, clicks and UI interactions. Also, Flet is theming and styling friendly, with support for light and dark mode, providing a visually pleasing user interface.

A dedicated Python module called Flet Audio enables text-to-speech (TTS) conversion and playing in real-time within Flet applications. This project uses it to create and play audio responses to text that is generated by artificial intelligence. Users can listen to responses in Tamil and English thanks to Flet Audio's multilingual capability. The user interface is kept responsive during the processing and playing of audio files since the audio functionality is implemented asynchronously. There is no need to keep pre-recorded files because Flet Audio's TTS engine produces speech dynamically. To enhance the user experience, a loading animation is also shown while the audio is being processed. Users can listen to the response when the play button appears after the audio is available.

Django is a web framework for Python that facilitates rapid development and pragmatic, clean design. It uses the Model-View-Template (MVT) architecture to promote the separation of business logic from the user interface. Django has database management, authentication and security built into it, making it a solid backend development choice. The system comes equipped with an ORM (Object-Relational Mapper) through which developers can operate databases with the help of Python rather than SQL, lowering database management complexity. Django is quite scalable and effectively manages a vast amount of data and user queries. Security is a key attribute of Django. Security is a core feature, with built-in protection against SQL injection, cross-site scripting (XSS) and cross-site request forgery (CSRF). In addition, Django also boasts a live community, comprehensive documentation and a modular framework that enables developers to construct and manage applications efficiently.

A strong and adaptable toolset for creating Web APIs in Django applications is the Django Rest Framework (DRF). In addition to offering authentication, permission management and response customisation, it makes data serialization simple. In this project, API endpoints for managing chat messages and AI responses are created using DRF. The framework facilitates JSON-based communication between the Flet frontend and the Django backend, making integration easier. Multiple endpoints can be efficiently managed with little code duplication thanks to DRF's support for class-based views and view sets. Additionally, it has integrated authentication features like token authentication that guarantee safe user interactions. DRF improves the application's scalability by simplifying the management of API versioning and pagination.

Groq API is a cloud-based platform that gives users access to complex big language models, including the LLaMA LLM model. In this project, the Groq API is utilized to create AI-powered replies to user inquiries. The API includes natural language processing (NLP) capabilities, which enable it to understand and answer to complicated questions. Groq's ability to handle multilingual queries is one of its most notable characteristics, making it appropriate for applications that require responses in both English and Tamil. The API is optimized for maximum speed, resulting in quick response times for chat conversations. It also offers contextual awareness, which means it can keep track of conversations and generate appropriate responses. Groq's interface with Django offers real-time AI-powered help, increasing the chatbot's interactive and informative capabilities.

MyMemory is an online translation API that offers reliable translations in a variety of languages, including Tamil and English. In this project, the Django backend uses the MyMemory API to translate AI-generated replies based on the user's language preference. The chatbot can easily switch between English and Tamil thanks to the automated translation process. When a user picks Tamil, the AI-generated English response is first cleaned to eliminate markdown formatting before being translated with MyMemory. The translated text is then routed back to the Flet UI for display and TTS conversion. This ensures that users receive responses in their selected language without requiring manual intervention. The incorporation of MyMemory improves the chatbot's accessibility, making it more usable for Tamil-speaking users.

In e-commerce and business, GenAI enables AI-driven customer support, product recommendations and marketing content generation. Platforms like Shopify and Amazon utilize AI to generate product descriptions, personalized recommendations and conversational AI support. In the entertainment industry, GenAI powers AI-driven scriptwriting, content recommendation systems and automated dubbing, as seen in Netflix's AI-powered translation and subtitle generation models. Similarly, in news and media, AI-generated journalism is used by organizations like The Washington Post and Bloomberg, where automated content generation models produce real-time financial reports and news summaries.

GenAI shows that it has both challenges and benefits, from legal and ethical concerns in government use to its ability to improve education and research. By addressing these issues, experts can ensure GenAI is used safely while still encouraging innovation. Working together across different fields and being transparent can help make GenAI a fair and sustainable tool for businesses and knowledge systems.

By implementing these enhancements, ARV-GPT will not only provide textual and audio-based assistance but also offer a more interactive and visually rich experience, making it a comprehensive AI-powered travel assistant to historical information, enhancing user engagement and making cultural exploration more immersive and informative.

III. PROPOSED METHODOLOGY

The ARV-GPT system is a virtual AI assistant meant to deliver precise information regarding Tamil Nadu's heritage places. It incorporates Flet (Frontend UI), Django Rest Framework (DRF) (Backend API), Groq API (AI processing with LLaMA LLM), TTS (Text-to-Speech) and multilingual translation (English & Tamil) to achieve an uninterrupted user experience. The system utilizes a sequential methodology, spanning from user authentication (login and registration) to token-based safe communication, querying, AI-generated responses, audio streaming in real time and frontend-backend.

The app begins with a signup and login feature to securely authenticate the users. Upon signup, the user enters information such as username, email and password on the Flet signup screen that sends an API request to the Django (/api/signup/) endpoint. Backend Django, fueled by the DRF's authentication process, securely saves the user credentials in the SQLite database. Thus the user can login after signup.

Upon login, the user inputs his credentials on the Flet login page, which are forwarded to the Django authentication API (/api/login/). Django verifies the credentials against the database and, upon successful validation, creates a DRF authentication token, which is forwarded to the frontend. The token is stored on the Flet session and utilized on all subsequent API requests, allowing secure access.

After the login, the Flet frontend sends all its requests with the authentication token added to the Authorization header. It is important for securing server-client communication. The chat API (/api/chat_ai/) needs the token for authenticating. Without a valid token, if any unauthorized request is made, Django blocks it and unauthorized access is not allowed.

© 2025 IJMRSET | Volume 8, Issue 3, March 2025| DOI: 10.15680/IJMRSET.2025.0803100

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Once authenticated, the user is redirected to the Flet-based chat interface, which is designed to provide an interactive and seamless experience. The interface consists of several key components, each with specific functionalities to enhance user engagement. The input field allows users to enter their queries, while the send button submits the message to the AI system for processing. The chat display area maintains the conversation history, ensuring users can view past interactions for reference. A language toggle switch enables users to switch between English and Tamil, ensuring multilingual support for a broader audience. The audio button provides real-time text-to-speech (TTS) functionality, allowing users to listen to AI-generated responses in their preferred language. To ensure a smooth user experience, an animated typing indicator appears while the system processes the query, simulating AI engagement and keeping users informed. The system also features a loading indicator for asynchronous audio generation, replacing the play button with a loading animation while the speech synthesis is in progress. When the user submits a query, the message is instantly displayed in the chat window and the frontend sends an asynchronous API request (/api/chat_ai/) to the Django backend. The backend processes the query using the LLaMA 3 (llama3-8b-8192-instant) model via the Groq API, ensuring fast and contextually relevant responses. If Tamil is selected, the system translates the response before displaying it. The UI follows a dynamic theme mode, automatically adapting to the system's light or dark mode, providing a visually appealing and accessible user experience. These features work together to create an intelligent and user-friendly virtual assistant, delivering real-time, multilingual and interactive assistance for Tamil Nadu's heritage site exploration.

To ensure ARV-GPT delivers relevant, accurate and well-structured responses while adhering to ethical AI principles, the system prompt is fine-tuned when making requests to the Groq API using the LLaMA 3 (llama3-8b-8192- instant) model. This tuning ensures that ARV-GPT remains focused on Tamil Nadu’s heritage sites, offering informative, concise and engaging responses about historical landmarks, architectural significance, cultural importance and practical travel details. The AI avoids technical jargon and instead provides user-friendly explanations while ensuring responses remain unbiased, non-political and free from controversial discussions. If a user query deviates from Tamil Nadu’s heritage context, ARV-GPT politely redirects the conversation to a relevant topic. The system is also designed to handle unclear questions by prompting users for clarification instead of making assumptions. Furthermore, when users toggle the language preference, ARV-GPT seamlessly translates responses into Tamil while maintaining clarity and correctness. Strict ethical safeguards prevent the generation of misleading, harmful, or biased content, ensuring the AI provides only factual and responsible information. Additionally, responses are structured in a way that enhances readability, engagement and user experience. By implementing these system prompt constraints, ARV-GPT not only ensures high-quality interactions but also aligns with its purpose as a reliable AI-powered virtual assistant for Tamil Nadu’s cultural and historical heritage.

When the user inputs a query in the chat interface, the chosen language from the toggle switch is what dictates the response processing path. If English is chosen, the query is directly passed to the Groq API with the LLaMA model and the response is received in English. When Tamil is chosen, the AI-responded English answer is initially translated to Tamil via the MyMemory API with a character limit of 500 for free-tier support. When the translated or the original response is received, a cleaning process is used to eliminate markdown formatting, special characters and redundant symbols in order to achieve clearness in both text output and audio conversion. The filtered text is subsequently shown on the chat window and synthesized in real-time by Flet-Audio. The process involves, temporarily, replacing the audio button with a loading indicator while generating the audio asynchronously. After the generation of the speech, the button is then reset to a play icon where the user can hear the AI-synthesized answer in the user's own chosen language.

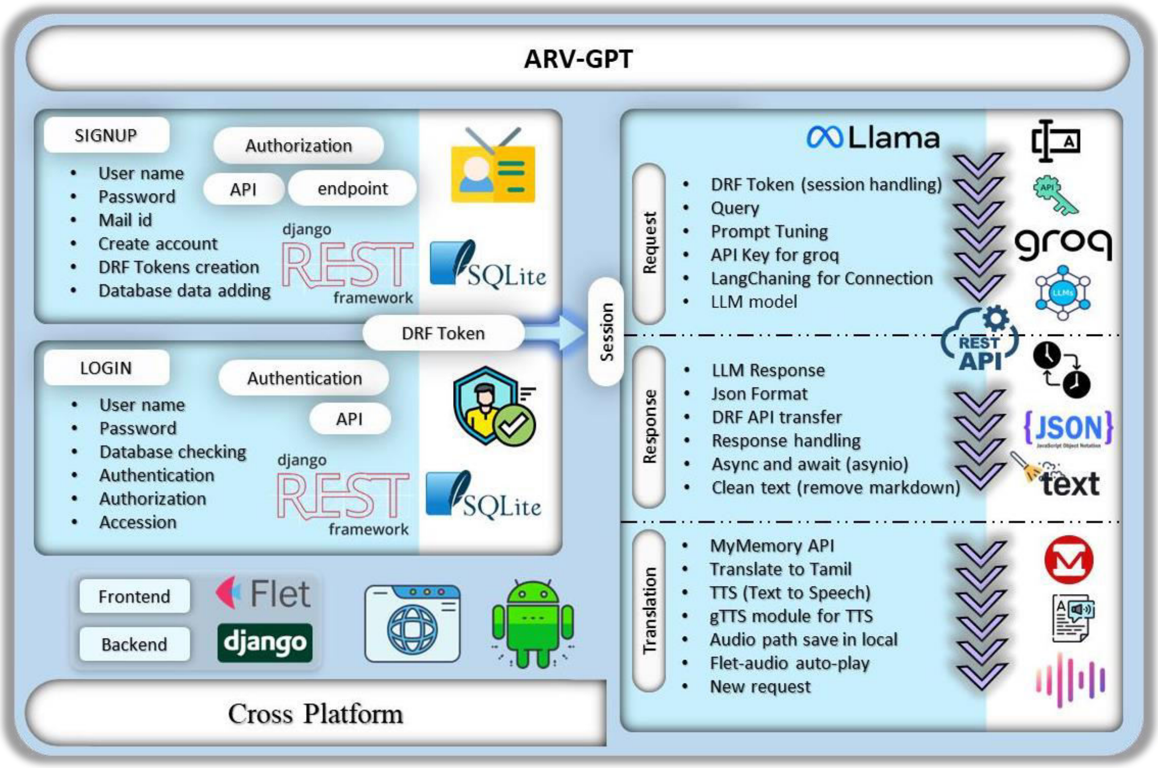


FIGURE 3.1 TECHNOLOGICAL ARCHITECTURE

ARV-GPT is a smart AI-powered chatbot designed to assist users in exploring Tamil Nadu's heritage sites with real-time, interactive and multilingual support. The system leverages Django Rest Framework (DRF) for backend processing and user authentication using token-based login. Once authenticated, users can engage with the chat interface built with Flet, which includes features such as an input field for queries, a chat display, a language toggle switch and an audio playback button. When a user submits a query, the request is sent to the Django backend, where the Groq API with the LLaMA-Instant model processes the input. If Tamil is selected, MyMemory API translates the response before being displayed. The system also ensures that markdown elements are removed from the text before converting it to speech using Flet-Audio for real-time playback. Asynchronous handling is used to enhance user experience, ensuring smooth UI transitions, quick response rendering and real-time audio generation without delays. Cloud hosting is enabled with PythonAnywhere for the backend and Firebase for the frontend, making ARV-GPT scalable and accessible. Future enhancements include integrating maps, augmented reality (AR) for virtual heritage site exploration and AI-driven personalization to refine user recommendations.

IV. TECHNOLOGIES USED

1. PYTHON DJANGO:

Django is a high-level Python web framework known for its speed, security and scalability, making it a preferred choice for building modern web applications. It follows the Model-View-Template (MVT) architecture, which enables developers to create structured and maintainable applications with minimal effort. One of its primary uses is web application development, where it provides built-in support for URL routing, database management, authentication and form handling, allowing developers to focus on business logic rather than repetitive coding tasks. Django is also widely used for API development, particularly with the Django Rest Framework (DRF), which simplifies the creation of RESTful APIs for mobile applications, AI-driven systems and cloud-based services.
A key advantage of Django is its security features, including built-in protection against SQL injection, cross- site

© 2025 IJMRSET | Volume 8, Issue 3, March 2025| DOI: 10.15680/IJMRSET.2025.0803100

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

scripting (XSS) and cross-site request forgery (CSRF). These security mechanisms make Django a reliable choice for handling sensitive user data and authentication systems. Additionally, Django's scalability and performance enable it to support large-scale applications, making it ideal for high-traffic websites and enterprise solutions.

2. LANGCHAIN:

LangChain is a powerful framework designed to build applications that utilize Large Language Models (LLMs) efficiently by integrating context management, data retrieval and decision-making capabilities. It simplifies the development of AI-driven applications by providing modular components for handling LLM chaining, memory persistence, API integration and external data connectivity. One of the key strengths of LangChain is its ability to combine multiple AI models and external data sources, allowing for context-aware responses and more accurate information retrieval.

LangChain works by structuring AI interactions into "chains", where multiple LLM calls are orchestrated in a logical sequence. This ensures that responses retain context across interactions, making it ideal for conversational AI, chatbots and AI-driven decision-making systems. A major advantage of LangChain is its retrieval-augmented generation (RAG) capability, which enables AI models to fetch real-world, up-to-date data from knowledge bases, APIs, or document repositories before generating responses. This significantly improves the accuracy and relevance of AI- generated content, making LangChain particularly useful for research assistants, virtual customer support, legal analysis and personalized recommendations.

Additionally, LangChain includes memory management features, allowing AI models to remember previous interactions and maintain conversational continuity, making interactions more dynamic and user-friendly. It seamlessly integrates with various database systems, vector stores (such as FAISS and Pinecone) and cloud storage solutions, enabling efficient knowledge retrieval and long-term AI training enhancements. Its flexibility extends to multi-agent AI orchestration, where different AI models collaborate within a single pipeline, optimizing workflow automation, content synthesis and AI-driven analytics.

LangChain's compatibility with frameworks like OpenAI, Hugging Face and Groq API makes it an essential tool for building intelligent, adaptive AI solutions. Whether used in chatbots, document summarization, real-time AI assistants, or data-driven insights, LangChain provides a scalable, modular and highly customizable framework for leveraging LLMs in practical applications. Its ability to connect AI with real-world data sources makes it a critical component for next-generation AI-driven automation.

On the Django backend, the system verifies the authentication token using DRF's authentication system. If valid, the request is processed further. The query is sent to the Groq API, which uses the LLaMA LLM model to generate an AI response. The response is initially produced in English. If Tamil is selected, Django translates the response using the MyMemory translation API, ensuring accurate multilingual support. The AI response is then cleaned to remove markdown formatting and special characters before being returned to the frontend.

3. FLET:

Flet is a Python-based framework that enables developers to create interactive web, desktop and mobile applications without requiring extensive frontend development skills. It is implemented on top of Flutter, guaranteeing seamless UI rendering and cross-platform support. In contrast to conventional web frameworks that demand the use of HTML, CSS and JavaScript, Flet eases the process of development through the ability of developers to develop interfaces using only Python. With its declarative UI paradigm, Flet enables users to define elements like buttons, text fields, images and layouts in an organized way, facilitating the creation of contemporary and visually stunning applications. The system also supports real-time updates and event-driven programming, allowing for smooth user interaction without such intricate state management.

One of the most important aspects of Flet is that it can be integrated with external APIs and backend services, which makes it perfect for developing dynamic applications like chatbots, dashboards and AI assistants. With its in-built state management and asynchronous execution features, Flet provides seamless communication between the frontend and backend, which improves performance and responsiveness. It also has support for a broad variety of UI components, such as interactive lists, containers, dialogs and progress indicators, which can be tailored to suit different application needs. Flet also offers an easy method of managing user authentication, form submission and data fetching, making it a great option for applications that need secure and effective user interaction.

Flet's cross-platform nature makes it a great fit for developers looking to create applications that operate effortlessly across various operating systems, such as Windows, macOS, Linux android, iOS and web browsers. This allows for reduced development time and effort, as a codebase can be shared across different platforms with minor adjustments. The framework's high-performance rendering engine guarantees that applications have a uniform look and feel across devices, making it perfect for developers who want to target a wide audience without having to contend with platform-specific intricacies. Another significant strength of Flet is its support for multimedia content, such as audio and video playback, which is especially useful for AI-based projects such as voice assistants, interactive narrative apps and learning tools.

With its real-time interaction support, developers can create applications that include live updates, which enhance the user experience by making it more interactive and engaging. The asynchronous event handling capability provides for smooth background processing so that applications stay responsive even while carrying out resource- intensive operations like AI inference or real-time translations. By merging the ease of Python with the strength of Flutter, Flet provides a smooth development process, enabling developers to build rich feature applications with ease. With integration of backend APIs, database systems and authentication methods, it further adds to its flexibility, making it adaptable to numerous use cases, ranging from chat apps to enterprise solutions.

With native UI components and a declarative programming paradigm, Flet enables developers to create modern, scalable and real-time applications without the need for advanced frontend development knowledge.

WHY DJANGO?

One of the main reasons to choose Django is its rapid development capabilities, as it follows the "batteries- included" philosophy, providing built-in features like authentication, database management, URL routing and security mechanisms, reducing the need for external dependencies. It follows the Model-View-Template (MVT) architecture, ensuring clean code structure and maintainability, making it easy for developers to build and scale applications efficiently.

Another major advantage of Django is its robust security features, which help protect applications from SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF) and other vulnerabilities. This makes Django a trusted framework for handling user authentication, sensitive data and enterprise applications.

Django is also highly scalable, making it suitable for both small projects and large-scale enterprise applications. It is used by high-traffic websites like Instagram, Pinterest and Mozilla, demonstrating its ability to handle massive amounts of data and concurrent users. Furthermore, Django supports asynchronous processing, enabling applications to handle real-time user interactions and background tasks efficiently. Its compatibility with cloud services like AWS, Google Cloud and PythonAnywhere makes deployment seamless. With its modular design, reusable components and strong community support, Django is an excellent choice for building secure, maintainable and scalable web applications.

WHY GROQ?

Groq is a high-performance AI inference engine designed to deliver ultra-fast processing speeds, low latency and efficient large language model (LLM) execution. One of the primary reasons for choosing Groq is its optimized AI acceleration architecture, which significantly outperforms traditional GPU-based processing when handling LLM inference tasks. Unlike conventional AI hardware that relies on parallel processing, Groq is built on a deterministic, single-core processing model, ensuring predictable and ultra-low-latency responses, making it ideal for real-time AI applications such as chatbots, virtual assistants and real-time data processing systems.

Groq's specialized AI compute engine is tailored for transformer-based models, enabling faster inference times compared to mainstream alternatives. This makes it highly suitable for applications requiring instant responses, such as customer support AI, real-time language translation, financial analytics and AI-driven decision-making systems. Additionally,

© 2025 IJMRSET | Volume 8, Issue 3, March 2025| DOI: 10.15680/IJMRSET.2025.0803100

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Groq’s cloud-based infrastructure allows seamless scalability, ensuring that AI applications can handle large volumes of requests without performance bottlenecks.

Another key reason for using Groq is its cost-efficiency and energy optimization. Compared to traditional AI inference solutions, Groq requires less power while maintaining high computational throughput, reducing operational costs for businesses deploying AI models at scale. The compatibility with major AI frameworks such as PyTorch and TensorFlow ensures that developers can easily integrate Groq into existing AI workflows. Overall, Groq is an excellent choice for real-time AI applications, offering a high-speed, cost-effective and scalable solution for LLM inference and AI-driven automation.

WHY FLET?
Flet was selected for this task because of its ease of use, cross-platform support and real-time interactive features, making it the best framework for creating an AI-based chatbot with multimedia capabilities. Because Flet makes it easy for developers to build interactive UI applications using Python without the requirement of much frontend expertise, it makes integration with the Django backend easy. This simplifies the development process without compromising the quality of user experience on different platforms, including desktop, mobile and web.

One of the important features of Flet here is that it is capable of handling user interaction smoothly. Flet's UI elements are utilized to create the chat interface with features such as input fields for the user, a send option to send the message and an area for displaying the chat history. Dynamically, there is a feature provided to change between English and Tamil, hence offering multilingual functionality. Also, the typing indicator increases user interaction by showing live feedback while waiting for the AI reply. Another notable aspect of Flet in this project is that it is integrated with `flet-audio`, which supports text-to-speech (TTS) functionality.

When users ask for an audio reply, the UI briefly replaces the audio button with a loading animation while the audio file is being created asynchronously. When prepared, the button reverts to a play icon, supporting seamless and intuitive user interaction. Flet's event-driven architecture guarantees efficient handling of API responses from the Django backend in the form of text-based and translated messages, making the chatbot interactive and responsive. By developing on Flet's capabilities, the project realizes an AI-driven, multilingual chatbot with real-time processing and voice-based support.

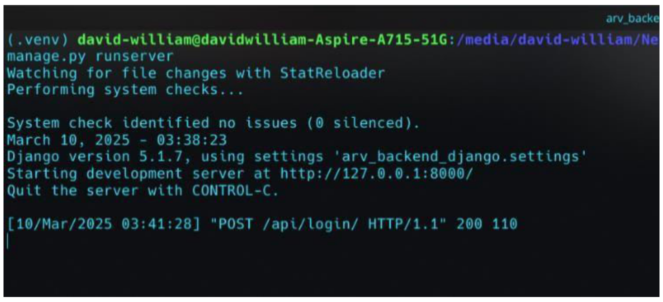
One of the major functionalities of Flet in this project is managing user interactions efficiently. The chatbot interface consists of various UI elements such as a text input field, a send button and a chat display area for conversation history. A language toggle switch enables users to select between English and Tamil, ensuring accessibility for diverse users. The chatbot also includes a typing indicator to provide real-time feedback, making the experience more interactive. The app theme adapts dynamically to the system’s dark or light mode, ensuring a visually appealing and user-friendly interface handle event-driven programming ensures that API calls to the Django have real-time and asynchronously run to avoid user interface freezing issue.

V. RESULT AND DISCUSSION

The ARV-GPT chatbot efficiently merges AI-driven natural language processing with real-time user engagement to deliver an intelligent virtual assistant to explore heritage destinations in Tamil Nadu. The combination of Django Rest Framework (DRF) as the backend, Flet for frontend UI, Groq API with the LLaMA model for AI-driven response and MyMemory for English-Tamil translation is successful. Users can input text to interact with the chatbot, get AI-driven responses, switch between languages and play text-to-speech (TTS) audio for a complete experience. The app provides a seamless and responsive user interface by using asynchronous operations to manage user requests and process responses without UI delay.

The chatbot provides highly context-specific and informative answers about Tamil Nadu's heritage places, with information on cultural importance, accessibility and visitor regulations. MyMemory translation integration provides easy communication in both English and Tamil, increasing reach. Flet-audio integration facilitates real-time TTS conversion, so users can listen to the AI response in their native language. Also, the theme adaptation functionality allows the UI to be synchronized with the dark or light mode of the system to improve user experience.

Django Server Initialization: The server starts on terminal, handling authentication and API requests.



```
                                                                        arv_backe
(.venv) david-william@davidwilliam-Aspire-A715-51G:/media/david-william/Ne
manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
March 10, 2025 - 03:38:23
Django version 5.1.7, using settings 'arv_backend_django.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.

[10/Mar/2025 03:41:28] "POST /api/login/ HTTP/1.1" 200 110
```

FIGURE 5.1 SERVER INTIALIZATION (DJANGO)

User Authentication: In the login page response of user details is user for the user authentication system is implemented using Django Rest Framework (DRF) with token-based authentication, where user credentials are validated against the SQLite database and a secure token is issued upon successful login to authenticate API requests.

User Query Processing: When the user enters a query and sends it, the system immediately shows the message while a typing indicator comes up, indicating AI response generation.
AI-Driven Response: The request is sent by the chatbot to the Django backend, which runs it through the Groq API with the LLaMA model and responds with a detailed answer.

Translation & Display: In case Tamil is chosen, the system translates the AI response with MyMemory before it is displayed in the chat interface.

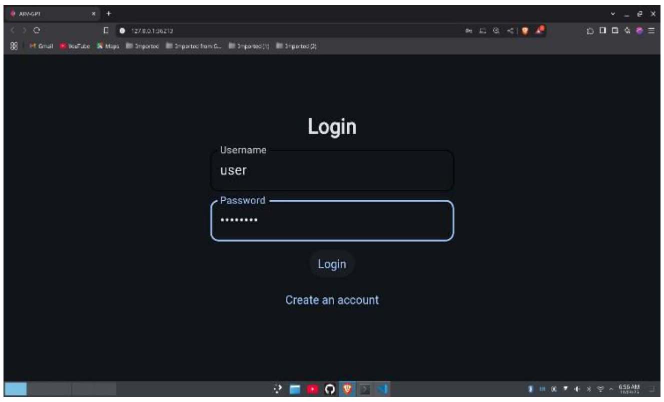


FIGURE 5.2 WEB LOGIN (FLET)

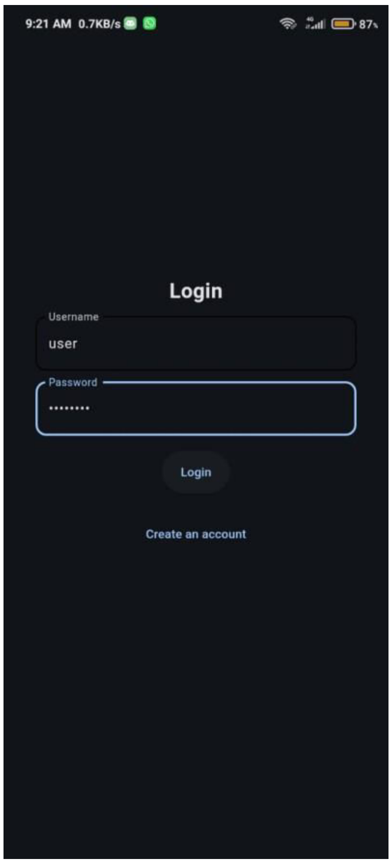
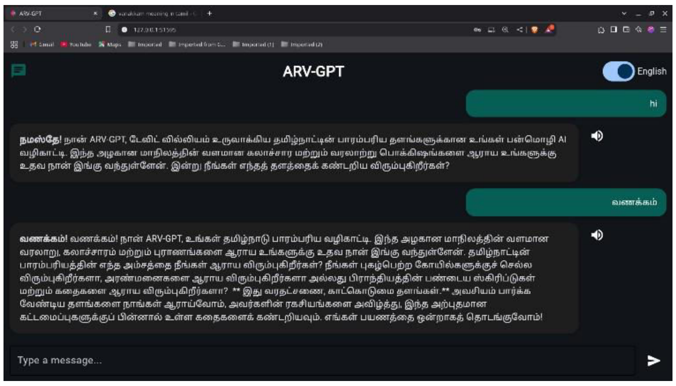


FIGURE 5.3 WEB CHAT (FLET)



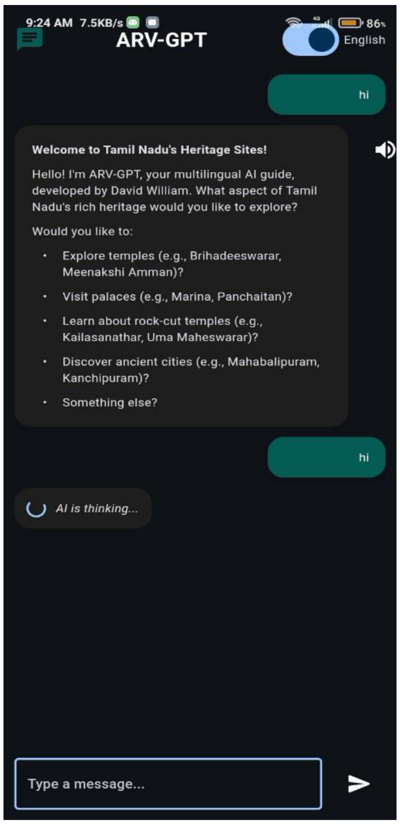
hi

நமஸ்தே! நான் ARV-GPT, டேவிட் வில்லியம் உருவாக்கிய தமிழ்நாட்டின் பாரம்பரிய தளங்களுக்கான உங்கள் பன்மொழி AI வழிகாட்டி. இந்த அழகான மாநிலத்தின் வளமான கலாச்சார மற்றும் வரலாற்று பொக்கிஷங்களை ஆராய உங்களுக்கு உதவ நான் இங்கு வந்துள்ளேன். இன்று நீங்கள் எந்தத் தளத்தைக் கண்டறிய விரும்புகிறீர்கள்?

வணக்கம்

வணக்கம்! வணக்கம்! நான் ARV-GPT, உங்கள் தமிழ்நாடு பாரம்பரிய வழிகாட்டி. இந்த அழகான மாநிலத்தின் வளமான வரலாறு, கலாச்சாரம் மற்றும் புராணங்களை ஆராய உங்களுக்கு உதவ நான் இங்கு வந்துள்ளேன். தமிழ்நாட்டின் பாரம்பரியத்தின் எந்த அம்சத்தை நீங்கள் ஆராய விரும்புகிறீர்கள்? நீங்கள் புகழ்பெற்ற கோயில்களுக்குச் செல்ல விரும்புகிறீர்களா, அரண்மனைகளை ஆராய விரும்புகிறீர்களா அல்லது பிராந்தியத்தின் பண்டைய ஸ்கிரிப்டுகள் மற்றும் கதைகளை ஆராய விரும்புகிறீர்களா? ** இது வரதட்சணை, காட்கொடுமை தளங்கள்.** அவசியம் பார்க்க வேண்டிய தளங்களை நாங்கள் ஆராய்வோம், அவர்களின் ரகசியங்களை அவிழ்த்து, இந்த அற்புதமான கட்டமைப்புகளுக்குப் பின்னால் உள்ள கதைகளைக் கண்டறியவும். எங்கள் பயணத்தை ஒன்றாகத் தொடங்குவோம்!

FIGURE 5.4 WEB CHAT (FLET)


9:24 AM 7.5KB/s

ARV-GPT

English

hi

Welcome to Tamil Nadu's Heritage Sites!

Hello! I'm ARV-GPT, your multilingual AI guide, developed by David William. What aspect of Tamil Nadu's rich heritage would you like to explore?

Would you like to:

- Explore temples (e.g., Brihadeeswarar, Meenakshi Amman)?
- Visit palaces (e.g., Marina, Panchaitan)?
- Learn about rock-cut temples (e.g., Kailasanathar, Uma Maheswarar)?
- Discover ancient cities (e.g., Mahabalipuram, Kanchipuram)?
- Something else?

hi

AI is thinking...

Type a message...

FIGURE 5.5 WEB CHAT (FLET)

On click Audio Button: the reply is stripped of markdown elements, translated into speech and played asynchronously. There is a loading animation on the button until ready.

Smooth UI Interaction: The query history is always present until logout and all features (language switching, sending questions and audio playing) function smoothly, providing a user-friendly and interactive experience.

VI. CONCLUSION

ARV-GPT effectively combines AI-powered natural language processing with real-time translation and text- to-speech functions to deliver an interactive virtual tour guide for heritage tourism. By using Django Rest Framework (DRF) as the backend, Groq API with the LLaMA model for AI-powered responses and Flet for a user-friendly frontend, the system provides a smooth user experience on web and mobile platforms. The addition of multilingual capabilities, especially English and Tamil, makes it accessible to a wider audience and real-time speech synthesis increases user interaction. Visitors can ask about Tamil Nadu's heritage places, such as history, architecture and tourist information, in text and voice. Users can switch between English and Tamil, getting the answer in their desired language, both in text and voice. AI assistant offers verbal explanations of cultural locations, taking the place of conventional tour guides. Travelers can pose advanced questions about travel directions, surrounding landmarks, or cultural meaning, with AI delivering contextual feedback. The application is offered on web and mobile platforms, allowing for convenience of access by tourists in transit. In future we will expand the accessibilities of ARV-GPT like as Map and GPS Integration, AI-Driven Voice Assistance, Virtual Tour Guide, Image integration, Augmented Reality (AR) Integration and In business terms ticket and booking for visitors.

REFERENCES

1. Interactive Language Learning Application using Artificial Intelligence. J. Linares Carrasquer 2025 Universitat Politècnica de València.
2. Comparativ Evaluation of Large Language Models for Text-to-SQL Parsing: A Case Study of Hotel Search Queries. D. Hardt 2024 Copenhagen Business School.
3. Benchmarking Public Large Language Models. V.M. Malode 2024 OPUS4 Repository.
4. Applying Large Language Model Analysis and Backend Web Services in Regulatory Technologies for Continuous Compliance Checks. J. Li; A. Maiti 2025 Future Internet Journal.
5. PyGen: A Collaborative Human-AI Approach to Python Package Creation. S. Barua; M. Rahman; M.J. Sadek;R. Islam 2024 arXiv Preprint.
6. Development of a Multi-Agent, LLM-Driven System to Enhance Human-Machine Interaction: Integrating DSPy with Modular Agentic Strategies and Logical Reasoning. Y. Mendoza Juan 2024 UPC Repository.
7. AI-Based Multiagent Approach for Requirements Elicitation and Analysis. M.A. Sami; M. Waseem; Z. Zhang;Z. Rasheed 2024 arXiv Preprint.
8. Privacy-Focused LLM for Local Data Processing: Implementing OLLAMA and RAG to Securely Query Personal Files in Closed Environments. A. Rodríguez Quiñones 2025 Open University of Catalonia Repository.
9. Walledeval: A Comprehensive Safety Evaluation Toolkit for Large Language Models. P. Gupta; L.Q. Yau;H.H. Low; I. Lee; H.M. Lim 2024 arXiv Preprint.
10. NLP-Driven Generative AI for Personalized Data. H. Ahmed; A.A. Tariq; M.S. Arham; H. Sattar 2024 NUST Repository.
11. Can We Trust Large Language Models Generated Code? A Framework for In-Context Learning, Security Patterns and Code Evaluations Across Diverse LLMs. A. Mohsin; H. Janicke; A. Wood; I.H. Sarker 2024 arXiv Preprint.
12. Large Language Models in Semantic Parsing for Log Analysis and Anomaly Detection. N. Sharma 2024 Norwegian University of Life Sciences (NMBU) Repository.
13. Large Language Model-Assisted Software Engineering: Prospects, Challenges and a Case Study. L. Belzner; T. Gabor; M. Wirsing 2023 Springer.