International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

OFFLINE PERSONAL ASSISTANT

SIDDHARTH T[1], SAI DHINAKAR S[2], SRIDHANWANTH S[3], PRIYADHARSHINI S[4] ,GUIDE Mrs.HEMALATHA[5]

DEPARTMENT B.Tech ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING SRI SHAKTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY

ABSTRACT :

Our application is designed to provide users with a customizable and private offline personal assistant powered by AI. The main goal is to enable users to create a tailored AI model that meets their specific needs, ensuring full privacy by functioning offline. This solution empowers users to control and personalize actions, tasks, and responses according to their preferences, offering enhanced usability, data security, and flexibility. The application aims to revolutionize how individuals interact with their devices, giving them a seamless and adaptable AI experience without compromising on privacy.

Keywords—offline, personal assistant, flexibility.

INTRODUCTION:

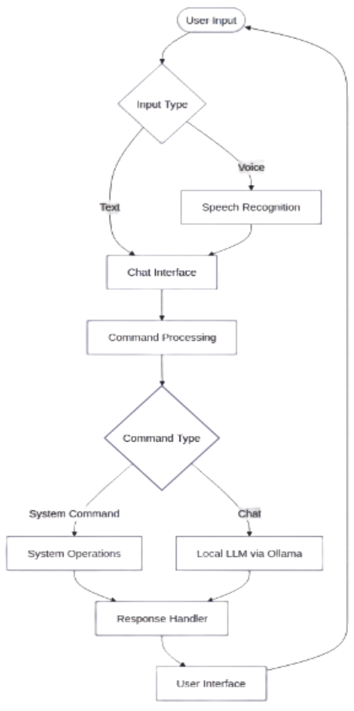
In today's fast-paced world, staying organized and managing daily tasks can be challenging. An offline personal assistant application is designed to help users simplify their lives by offering a range of features without requiring an internet connection. This project aims to provide a user- friendly, secure, and reliable tool to manage schedules, set reminders, take notes, and perform other essential tasks efficiently. By working offline, the assistant ensures privacy and accessibility at all times, making it a practical solution for personal organization.

In an increasingly digital world, personal assistants powered by AI have become integral to simplifying daily tasks, managing schedules, and providing information on demand. However, most current solutions rely heavily on cloud-based systems, raising concerns about data privacy, security, and internet dependency.

To address these challenges, offline personal assistants are emerging as a transformative solution. Unlike cloud- dependent systems, offline personal assistants operate directly on the user's device, ensuring data privacy and eliminating the need for constant internet connectivity. These systems provide a highly customizable and secure environment where users can interact with their assistant confidently, knowing their personal information remains under their control.

This innovative approach enables a tailored experience, allowing users to define specific actions, workflows, and preferences, making the assistant adaptable to individual needs. As a result, offline personal assistants offer a blend of convenience, personalization, and privacy, reshaping the future of digital interaction.

METHODOLOGY:



Creating an offline personal assistant involves developing a system that can function independently of internet connectivity while providing various services.
Above is a methodology to design and implement such an assistant

REQUIREMENT ANALYSIS:

The goal of **DORE AI** is to create a fully offline personal AI assistant that prioritizes user privacy, customizability, and efficient performance without relying on cloud-based systems. This assistant must be capable of handling tasks such as voice command recognition, task management, file and system operations, application control, web search integration, and text extraction from images. These functionalities are essential to ensure that the assistant can manage user needs effectively and enhance productivity. Alongside these functional requirements, non-functional requirements like privacy, performance optimization, and customizability are critical. User data must be processed locally to ensure confidentiality, while the system must be optimized to run efficiently on devices with limited resources. Customizability allows users to integrate their preferred Large Language Models (LLMs) and tailor the assistant to their specific needs. Additionally, the platform must be scalable to support future plugins and extensions for added functionality.

SYSTEM ANALYSIS:

To meet these requirements, DORE AI's system must seamlessly integrate components for voice processing, task execution, and system interaction while maintaining a modular and scalable architecture. At the core, the system will feature a voice recognition module that converts user speech into text commands using offline tools such as Vosk or FasterWhisper. These commands are processed through a command parser that interfaces with various system functions, enabling task automation, file management, and system configuration adjustments. An AI integration framework, powered by Ollama, allows the use of local LLMs for handling complex queries and providing intelligent responses. The system also includes an optical character recognition (OCR) module using Tesseract to enable text extraction from images. Finally, the user interface will provide an intuitive and accessible way for users to interact with and configure the assistant. The system design faces challenges such as optimizing resource usage to support complex AI models on local hardware and ensuring accurate processing of voice and text inputs without cloud support. Scalability is another critical aspect, as the framework must accommodate future feature additions without requiring extensive rework of the core system.

TECHNOLOGICAL SELECTION:

Python was selected as the primary programming language for **DORE AI** due to its robust library ecosystem, simplicity, and compatibility with machine learning tools. For voice recognition, Vosk and FasterWhisper were chosen for their offline capabilities and low resource consumption. The integration of Large Language Models is facilitated by Ollama, which allows for customizable, locally running models to enhance the assistant's NLP capabilities. Tesseract OCR provides reliable text extraction from images while operating entirely offline.The system and file operations will be executed using Python's OS and subprocess libraries, ensuring comprehensive control over system-level tasks. To create the user interface, frameworks like PyQt or Tkinter will be used to deliver an interactive experience. The system's plugin support is designed using dynamic Python imports, enabling the seamless addition of new features without altering the core codebase.To optimize resource usage, libraries like NumPy and PyTorch will be employed for efficient computation and model execution, while tools like Joblib or Asyncio will handle asynchronous task management. Testing and deployment will rely on PyTest for quality assurance and PyInstaller to package the application into a standalone executable for Windows, ensuring ease of distribution and use.By integrating these carefully selected technologies and frameworks, DORE AI is positioned to deliver a secure, efficient, and customizable offline personal assistant, catering to the growing demand for privacy-focused and reliable AI solutions.

DEVELOPMENT:

Backend Development: Create the logic for managing task s, reminders, and notes.
Frontend Development: Build a simple, intuitive interface for user interaction.

TESTING:

Test the app for functionality, usability, and data security. Ensure all features work smoothly without an internet connection.

DEPLOYMENT:

Package the app for use on target devices and provide installation instructions.

CORE ARCHITECTURE:

Operating System: Choose a compatible OS (Windows).
Hardware Considerations: Ensure sufficient processing power, storage, and memory for running offline tasks

a.Modular Design
Break the system into components like speech recognition, text-to-speech, natural language processing (NLP), and task management.

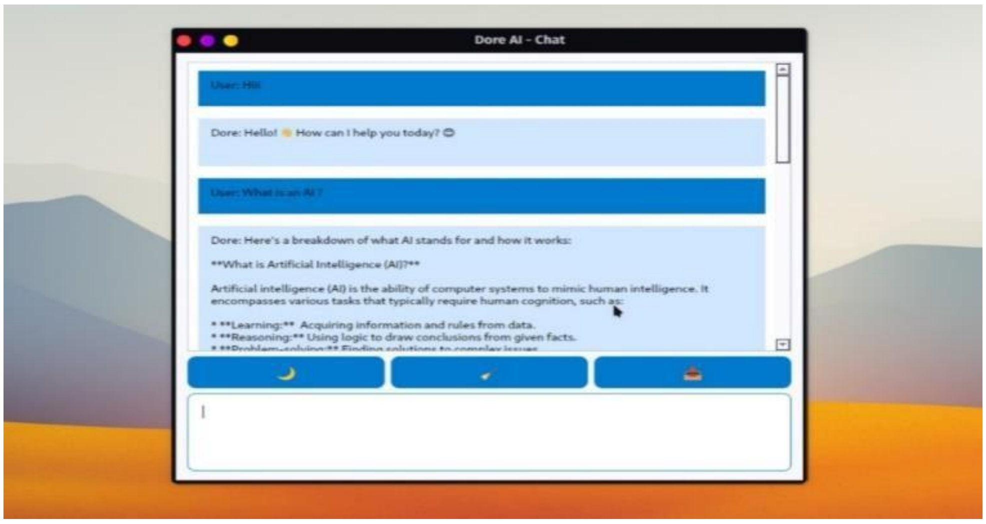
Use APIs or libraries that work offline.
b.Database Management
Use a local database (SQLite or TinyDB) to store user data securely.
Implement efficient indexing for fast data retrieval.

KEY COMPONENTS:

a.Speech Recognition
Use offline libraries like Vosk, CMU Sphinx, or DeepSpeech for speech-to-text conversion.
b.Natural Language Processing
Use lightweight NLP libraries like spaCy, rasa NLU, or custom rule-based systems for intent recognition.
Train models locally for specific tasks.
c.Text-to-Speech
Implement offline TTS systems like eSpeak, Festival, or MaryTTS for audio responses.
d.Task Scheduler
Create modules for setting alarms, reminders, and to-do lists using local time-based triggers.
e.File Management
Provide functionalities to search, open, and manage files locally using OS file handling.
f.Local Knowledge Base
Build a local repository of FAQs, dictionaries, or encyclopedic information for reference..
5. User Interface
Command-Line Interface (CLI) for simplicity.
Graphical User Interface (GUI) using frameworks like PyQt, Tkinter, or Electron.
Voice interface for hands-free operations.

FEATURES:

The offline personal assistant project successfully achieved its goal of providing a reliable, user-friendly application to manage daily tasks without requiring an internet connection. The following outcomes were observed:
CORE FEATURES IMPLEMENTED:
Task management and reminders. Note-taking and local data storage.
Easy access to all functionalities offline.
PRIVACY AND SECURITY:
Data is stored locally, ensuring user privacy.
No internet connectivity eliminates risks of external data breaches.
USER FRIENDLY INTERFACE:
Simple and intuitive design for effortless navigation. Features are easy to access and use for all age groups.
PERFORMANCE:
Smooth and fast performance without dependency on external servers.
Minimal resource usage, making it compatible with a wide range of devices.
Overall, the project delivered an effective solution for personal organization, helping users stay productive and organized even in offline environments.
VOICE INTERACTION:
Ability to recognize and process spoken commands offline using Vosk or CMU Sphinx.
Provides responses via text-to-speech using tools like eSpeak or pyttsx3.
TIME MANAGEMENT:
Set alarms, reminders, and timers offline.
To-do list and task prioritization stored locally in SQLite. LOCAL SEARCH AND FILE
HANDLING:
Search files on the local system using keywords or file types.
Open, delete, or move files with voice or typed commands.
OFFLINE KNOWLEDGE BASE :
Look up FAQs, encyclopedic entries, or other preloaded information without internet access.
Perform basic calculations and conversions offline.
USER FRIENDLY INTERFACE:
Voice-based interaction for hands-free usage.
Graphical interface built using PyQt or Tkinter for manual commands.



User: Hi!

Dore: Hello! 👋 How can I help you today? 😊

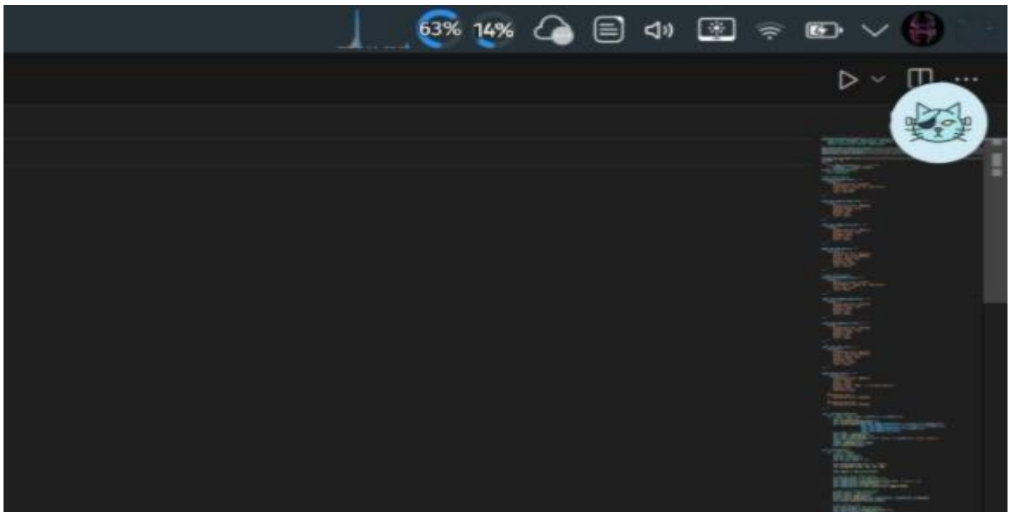
User: What is an AI ?

Dore: Here's a breakdown of what AI stands for and how it works:

What is Artificial Intelligence (AI)?

Artificial intelligence (AI) is the ability of computer systems to mimic human intelligence. It encompasses various tasks that typically require human cognition, such as:

* **Learning:** Acquiring information and rules from data.
* **Reasoning:** Using logic to draw conclusions from given facts.
* **Problem-solving:** Finding solutions to complex issues



CONCLUSION:

The offline personal assistant successfully demonstrates a privacy-focused and reliable solution for task management and voice interaction without internet dependency. While it offers robust functionality and security, it is limited in dynamic features compared to online systems. This project highlights the potential for secure, customizable assistants, ideal for users prioritizing data privacy and offline capabilities

REFERENCES:

[1]https://academy.networkchuck.com/blog/local-ai-voice-assistant
[2]https://medium.com/@lawrenceteixeira/revolutionizing-corporate-ai-with-ollama-how-local-llms-boost-privacy-efficiency-and-cost-52757390bf26