**FRIDAY : AI- ASSISTANT**

*Project Progress Report – I*

**BACHELOR OF TECHNOLOGY**

## **IN**

**COMPUTER SCIENCE AND ENGINEERING**



**Submitted to:**

**Dr. Abhay Bhatia**

**(Project Coordinator)**

**Associate Professor, Department of Computer Science and Engineering**

**Guided By:**

**Mr. Nitesh Kumar**

**Assistant Professor, Department of Computer Science and Engineering**

**Submitted by:**

Sparsh Kumar (210240101104)

Vikas Kumar (210240101118)

**Project Progress Report**

**(Jan - May 2025)**

**Progress Report: AI - ASSISTANT**

### **1. Project Overview**

The objective of **FRIDAY – My Personal AI Assistant** is:

To develop a smart, voice-controlled, multimodal AI assistant that seamlessly integrates advanced AI models and APIs to automate daily tasks, offer real-time answers, control devices and applications, and provide personalized user interactions using natural language.

**Key goals within that objective:**

1. **Voice Interaction**: Enable natural, voice-activated communication using a custom wake word (“Hey FRIDAY”).
2. **Real-Time Intelligence**: Use Grok and Google APIs for real-time data access, search, and reasoning.
3. **Device & App Control**: Automate opening/closing apps, accessing websites, and controlling hardware.
4. **Image & Text Generation**: Leverage Hugging Face APIs for generating images and NLP-based responses.
5. **Personalisation & Context**: Maintain conversation history and personalize responses based on user behaviour.
6. **Future Expandability**: Plan for smart home integration, spatial awareness, and multi-device compatibility.

### **2. Completed Tasks**

#### **2.1 Requirements Gathering**

* Conducted market research and competitor analysis.
* Identified user personas and functional requirements.
* Created a feature list and project roadmap.

#### **2.2 UI/UX Design of FRIDAY – My Personal AI Assistant** The UI/UX design of FRIDAY focuses on delivering a *sophisticated, voice-first interface* with optional visual elements, designed for efficiency, elegance, and futuristic aesthetics. Inspired by popular fictional AIs (like J.A.R.V.I.S), FRIDAY’s design philosophy revolves around minimal interaction effort with maximum functionality.

#### **🎨 Design Goals**

#### Minimalist & Modern Aesthetic: Clean layout, dark-themed interface with glowing neon accents (inspired by sci-fi UIs).

#### Voice-Centric Experience: Users primarily interact via voice commands; UI acts as a visual feedback companion.

#### Responsive & Adaptive: Interface scales beautifully across desktops, tablets, and other smart displays.

#### Emotionally Intuitive: UI elements like waveform animations, real-time speech bubbles, and expression-like feedback add “life” to the assistant.

#### **🧩 Key UI Components**

| **Component** | **Description** |
| --- | --- |
| **Wake Interface** | Animated circular or waveform visualizer that activates on “Hey FRIDAY”. |
| **Command Overlay** | Semi-transparent floating overlay displaying user command & FRIDAY’s response in real-time. |
| **Dashboard (Optional Visual UI)** | A glass morphism-based panel showing recent tasks, device control buttons, weather, reminders, etc. |
| **Settings Panel** | Voice-customization, integrations, smart device linking, and themes. |
| **Voice Transcript History** | Scrollable chat-like window for reviewing past interactions. |
| **Image Generator Output Window** | When generating images (via HuggingFace), a modal/lightbox displays output in high-res. |

#### **🔧 UI/UX Technologies Used**

#### **Frontend**: Tailwind CSS, HTML5, and minimal JavaScript/React if visual dashboard is enabled.

#### **Voice Feedback**: Web Speech API or Python TTS libraries for auditory responses.

#### **Animations**: CSS animations for waveform; Lottie/Framer Motion for micro-interactions.

#### **Design Inspiration**: Sci-Fi interfaces, Iron Man HUDs, Google Assistant UI.

#### **🤖 Example Interaction Flow**

#### User says "Hey FRIDAY, open Spotify and play my playlist."

#### FRIDAY activates with a glowing ring animation + soft TTS reply: *“Opening Spotify and playing your Chill Mix.”*

#### Command appears briefly as an overlay for confirmation.

#### Spotify opens, and the UI dims into standby mode.

#### **2.4 Backend Development**

| **Layer** | **Technology** |
| --- | --- |
| **Language** | Python 3.10.10 |
| **Frameworks** | FastAPI / Flask (for HTTP routes) |
| **AI Integrations** | Grok API, Google APIs, HuggingFace Transformers |
| **Device Control** | pyautogui, os, subprocess, pyttsx3 |
| **Speech Processing** | speech\_recognition, pyaudio, Google Speech-to-Text |
| **Text-to-Speech (TTS)** | pyttsx3, ElevenLabs, or Google TTS |
| **Scheduling / Background Tasks** | APScheduler or asyncio tasks |
| **Storage (if any)** | JSON files / SQLite (lightweight state saving) |
| **Environment Management** | .env + dotenv for managing keys and settings |
| **Testing** | pytest, unittest, and manual CLI interactions |

**2.5 User Authentication**

* Developed user login, registration, and session management.
* Implemented JWT-based authentication.
* Role-based access for customers and admins established.

### **3. Ongoing Tasks**

**📅 Ongoing Tasks (January – March 2025) :** These are tasks that were initiated and either completed or actively developed during the initial development phase.

| **Task** | **Description** |
| --- | --- |
| **✅ Project Setup & Environment** | Initialized using Python 3.10.10, with VS Code and environment configuration for external APIs. |
| **✅ Wake Word & Voice Command Recognition** | Implemented custom wake word "Hey FRIDAY" and real-time voice input parsing using Google STT. |
| **✅ NLP & Intent Parsing** | Used Google NLP and Grok integration to understand user queries and determine intent. |
| **✅ Grok Integration** | Connected FRIDAY to Grok-1.5V for real-time reasoning, chat capabilities, and multimodal support. |
| **✅ Google API Integration** | Used Google Cloud APIs for real-time data (weather, news, search, maps). |
| **✅ Device & App Control** | Added modules for monitoring open apps and controlling them (e.g., launching browsers, applications). |
| **✅ Natural TTS & Audio Output** | Built text-to-speech engine with emotional inflection and feedback using pyttsx3 or alternatives. |
| **✅ Noise Filtering & Stability Enhancements** | Implemented background noise filters and improved response reliability. |
| **⚙️ Image Generation (Partially Implemented)** | HuggingFace image generation API setup completed; integration with voice input partially tested. |

### **4. Upcoming Tasks**

**Upcoming Tasks (April – May 2025) :** These tasks represent the next phase of development, aimed at personalization, expansion, and advanced feature integration.

| **Planned Task** | **Objective** |
| --- | --- |
| **🔜 Contextual Memory Engine** | Allow FRIDAY to recall past conversations, personalize responses, and track user preferences. |
| **🔜 Smart Home Integration** | Integrate APIs to control smart devices (lights, thermostats, etc.) via voice commands. |
| **🔜 Spatial Awareness & CV Integration** | Use computer vision to identify user gestures, presence, or environment objects for contextual interaction. |
| **🔜 Personalisation Layer** | Enable FRIDAY to adjust tone, preferences, and behaviour per user (voice style, preferred tools). |
| **🔜 Unified Dashboard Interface** | Optional GUI for displaying responses, task lists, image output, and contextual cards. |
| **🔜 Performance Optimization** | Enhance latency, memory efficiency, and concurrency (async tasks, rate-limiting). |
| **🔜 Testing & Debugging** | Unit and integration testing, end-to-end validation in real scenarios. |
| **🔜 Deployment & Documentation** | Prepare final production-ready version, with detailed documentation and setup scripts. |

### **5. Challenges Faced**

**1. Latency in Voice Response Handling:** Ensuring near real-time feedback after receiving a voice command was difficult due to API call delays and local processing.

**2. Multi-API Synchronization:** Integrating multiple APIs (Grok, Google, HuggingFace) smoothly without blocking each other required careful async management.

**3. Context Retention in Conversations:** Maintaining coherent multi-turn conversations while handling diverse queries posed difficulties in managing session-based memory.

**4. Noise & Audio Quality Variance:** Voice recognition accuracy dropped in noisy environments, making robust noise filtering and microphone input processing critical.

**5. System-Level App Control Restrictions :** Accessing and automating hardware-level controls (like launching apps or adjusting system settings) faced OS permission limitations and required safe scripting.

### **6. Timeline Overview**

| **Task** | **Status** | **Expected Completion** |
| --- | --- | --- |
| UI/UX Design | ✅ Done | – |
| Voice Command Recognition | ✅ Done | – |
| Backend Core Setup | ✅ Done | – |
| Grok & Google API Integration | ✅ Done | – |
| Image Generation Module | ⏳ In Progress | 10 April 2025 |
| Contextual Memory Engine | ⏳ In Progress | 20 April 2025 |
| Smart Home Integration | ⏳ Upcoming | 28 April 2025 |
| Spatial Awareness Module | ⏳ Upcoming | 5 May 2025 |
| Personalisation Layer | ⏳ Upcoming | 10 May 2025 |
| Final Testing & Deployment | ⏳ Upcoming | 18 May 2025 |

### **7. Conclusion**

The development of *FRIDAY – My Personal AI Assistant* is progressing well and remains on track with the defined schedule. Core functionalities including voice command recognition, backend setup, and AI integrations with Grok and Google have been successfully implemented. Current efforts are focused on enhancing personalization, integrating image generation, and enabling contextual memory. Upcoming milestones include smart home integration, spatial awareness, and final testing. The collaboration between design, development, and AI modules has been efficient, ensuring a cohesive and intelligent user experience. FRIDAY is well-positioned for a stable beta release within the upcoming evaluation phase.

|  |  |
| --- | --- |
| ***Report prepared by:***  **Sparsh Kumar 210240101104**  **Vikas Kumar 210240101118** | ***Report approved by:***  **Dr. Abhay Bhatia**  **(Project Coordinator)**  May, 2025 |

# **1. Project identification**

|  |  |
| --- | --- |
| **Project title** | **FRIDAY AI-ASSISTANT** |
| **Group members** | **Member Name and Roll No. Role**   |  |  |  | | --- | --- | --- | | 1. | Sparsh Kumar(210240101104) | AI-Architect | | 2. | Vikas Kumar(210240101118) | Frontend, UI/UX | |
| **Technical advisor(s)** | **Project Coordinator:**   |  |  | | --- | --- | | Name: | Dr. Abhay Bhatia | | Designation: | Associate Professor |  **Project Guide:**   |  |  | | --- | --- | | Name: | Mr. Nitesh Kumar | | Designation: | Assistant Professor | |

## **CERTIFICATE**

|  |  |
| --- | --- |
| “This is to certify that the project work until evaluation held on **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**, titled as stated in *Sec. 1.1*, executed (as till date) by the students’ group mentioned in *Sec. 1.2*, has been found satisfactory and every section of this report is reflecting the same.” | *(Signature of project coordinator & date)* |

|  |  |
| --- | --- |
| **Thematic area(s)** | ☐ Research ☐ S/W Development ☐ Industry Automation  ☐ Institute Automation  ☐ Other *(please specify)*:  **Sub-areas** *(optional)*:  ☐ Mobile Programming ☐ Web Programming ☐ HCI  ☐ Data Mining ☐ Networking ☐ Image Processing  ☐AI /ML  ☐ Other *(please specify)*: |
| **Keywords** |  |
| **Utilization scope** | ☐ Domestic ☐ Commercial ☐ Industrial ☐ Scientific  ☐ Global ☐ National ☐ State ☐ District  ☐ GNDEC ☐ University ☐ External ☐ Sponsor  ☐ Other *(please specify)*: |
| **Major task(s)** *(At least one should be checked)* | ☐ Modeling ☐ S/W Designing ☐ Fabrication ☐ Testing  ☐ Validation ☐ Optimization ☐ Consultancy document  ☐ Physical tool(s) development ☐ Software development  ☐ Research & Development  ☐ Others *(please specify)*: |
| **Software packages, tools and programming languages** | ☐ Python ☐ VC++ ☐ C/C++ ☐ C#  ☐ Java ☐ PHP ☐ ASP/.NET ☐ OpenCV  ☐ AutoCAD ☐ MongoDB ☐ JAVA Script ☐ VB  ☐ MATLAB ☐ Android ☐ Oracle ☐ SAP  ☐ MySQL ☐ SQL  ☐ Mobile phone development *(please specify)*:  ☐ Others *(please specify)*: |

# **2. Relevant study material**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Undergrad. courses** | |  |  | | --- | --- | |  | **Course code and title** | | 1. | Web Development (Frontend & Backend) | | 2. | Artificial Intelligence & Machine Learning | | 3. | Natural Language Processing | | 4. | Software Engineering | | 5. | Human-Computer Interaction (HCI) and UI/UX Design | | 6. | Python Programming | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Standards and databases***(Must be easily accessible. Add more rows if required.)* | |  |  | | --- | --- | |  | **Name, purpose, source and other details** | | 1. | Google Cloud Speech-to-Text API | | 2. | Local State Management (SQLite / JSON-based) | | 3. | HuggingFace Transformers & Diffusers | | 4. | Grok API Standards (Grok-1.5V) | |
| **Online / web resources** | |  |  | | --- | --- | |  | **URL of specific web page** | | 1. | **Grok API Documentation**  **Website:** <https://x.ai> | | 2. | **Google Cloud Speech-to-Text API Docs** **Website:** <https://cloud.google.com/speech-to-text/docs> | | 3. | **HuggingFace Transformers Documentation** **Website:** <https://huggingface.co/docs/transformers> | | 4. | **Python SpeechRecognition Library Docs** **Website:** <https://pypi.org/project/SpeechRecognition/> | |

|  |
| --- |
| 1. **Objective/ Scope** |
| *The objective of this project is to develop* ***FRIDAY – a robust, intelligent, and voice-controlled personal AI assistant*** *that seamlessly integrates advanced natural language understanding, real-time web access, and system-level automation. FRIDAY is designed to function as a context-aware assistant capable of executing spoken commands, answering queries, generating images, and managing applications, offering users an efficient and futuristic interaction experience through multimodal communication.*  *The system primarily serves* ***individual users*** *by enabling them to perform everyday tasks such as opening apps, browsing websites, accessing live information (weather, news, maps), and even generating AI images or engaging in intelligent conversations—all through voice interaction. A key component of the system is its* ***integration with multiple APIs****, including Grok (for reasoning and multimodal chat), Google Cloud APIs (for speech recognition and web search), and HuggingFace (for NLP and image generation), ensuring rich and responsive capabilities.*  *The scope of this project includes complete* ***backend development using Python****, API integration for real-time data handling and content generation, and a* ***lightweight voice-activated UI/UX design*** *that prioritizes user comfort, accessibility, and speed. The assistant features key modules such as a wake-word activation system ("Hey FRIDAY"), command parsing engine, voice-to-text and text-to-speech pipelines, context memory handler, and a visual dashboard for displaying responses and outputs when needed.*  *FRIDAY also emphasizes* ***customization and scalability****, offering support for context retention, emotion-inflected TTS, and future smart home integrations. The backend logic has been built to support modular upgrades including advanced personalization, spatial awareness via computer vision, and multi-device compatibility for desktop and smart environments.*  *Furthermore, the assistant is equipped with* ***noise filtering algorithms****, a resilient response architecture, and a conversational personality layer that ensures continuity and relatability in long-term usage. Real-time performance optimization is at the core of the design, ensuring sub-second latency, asynchronous task execution, and secure API usage through environment-based credential management.*  *The project has been architected with extensibility in mind, paving the way for* ***upcoming enhancements*** *such as smart device control, multi-language support, gesture recognition, and integration with third-party productivity tools. By focusing on intuitive voice-first interactions, intelligent automation, and scalable architecture, FRIDAY aims to deliver a highly efficient, futuristic, and personalized AI assistant experience that blends seamlessly into users’ digital lives.* |

|  |
| --- |
| **5. Expected Output** |
| 1. ***Project Initialization***  * *Initialized the project using Python 3.10.10 with a modular folder structure.* * *Configured development environment using VS Code and .env files for managing sensitive API keys.* * *Installed essential libraries: speechrecognition, pyttsx3, requests, pyautogui, and asyncio.* * *Created GitHub repository and initialized version control with .gitignore and documentation base.* * *Setup asynchronous FastAPI backend for API routing and integration handling.*  1. ***Voice Command Interface***  * *Implemented wake word detection system using keyword spotting.* * *Developed microphone listener to capture and transcribe voice commands via Google STT.* * *Integrated command logger to visualize user input in terminal/debug mode.* * *Optimized latency by handling command recognition asynchronously.* * *Added safety timeout and fallback responses for unclear input.*  1. ***Grok & Google API Integration***  * *Successfully integrated with Grok-1.5V for AI-driven multimodal conversations.* * *Enabled real-time response capability using Grok’s reasoning engine.* * *Added Google Search and Cloud NLP API to process factual questions and commands.* * *Parsed natural language queries and structured them for downstream logic.* * *Stored API keys securely using .env and dotenv.*  1. ***Text-to-Speech Output (TTS)***  * *Implemented pyttsx3 for local text-to-speech engine.* * *Added emotional tone modulation and voice rate customization.* * *Enabled speech confirmation after every executed command.* * *Provided fallback to console output if audio fails.*  1. ***Image Generation Module***  * *Connected to HuggingFace image generation APIs (e.g., Stable Diffusion).* * *Designed voice-triggered prompt generator to describe images naturally.* * *Added functionality to display and save generated images.* * *Tested compatibility across low and high-resolution formats.*  1. ***Desktop & App Control***  * *Implemented system-level automation using pyautogui and os.* * *Enabled basic commands: "Open Chrome", "Launch Notepad", "Close Spotify", etc.* * *Mapped voice instructions to system functions with a routing engine.* * *Provided visual and auditory confirmation after successful execution.* * *Added support for custom app configuration via JSON settings.*  1. ***Audio Processing & Noise Filtering***  * *Integrated noise reduction logic using threshold-based filtering.* * *Handled multiple microphone input types.* * *Improved recognition accuracy in non-ideal acoustic environments.* * *Prevented accidental trigger using confidence threshold settings.*  1. ***Command Routing & Dispatcher***  * *Created a dynamic routing system to delegate tasks to correct modules (e.g., browser, image, API).* * *Built error-handling routines for unrecognized commands or missing modules.* * *Included debug mode for testing API call outcomes.* * *Future support planned for multi-command chaining and task queueing.* |

# 

# **Methodology**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **S. No.** | **List of project milestones** | **Deliverable(s)** | **Expected number of days to complete** | **Percent Completed** | | --- | --- | --- | --- | --- | | 1. | Project Initialization | Python environment setup, folder structure, Git version control | 1 | 100% | | 2. | Wake Word Detection | "Hey FRIDAY" wake word listener with low-latency response | 2 | 100% | | 3. | Voice Command Parsing | Speech-to-text via Google STT + command intent recognition | 3 | 100% | | 4. | Grok Integration | Real-time conversational AI and image generation | 3 | 100% | | 5. | Google API Integration | Weather, search, map access via Google API | 2 | 100% | | 6. | System App Control | Launching and closing desktop applications via voice | 3 | 100% | | 7. | TTS Output | Voice responses using pyttsx3 with modulation | 2 | 100% | | 8. | HuggingFace Image Generator | Image generation from text prompts via HuggingFace API | 2 | 80% | | 9. | Noise Filtering | Noise suppression and threshold calibration for clean input | 2 | 100% | | 10. | Contextual Memory | Memory for maintaining user context and prior interactions | 3 | 30% | | 11. | Smart Home Integration | IoT device control via external APIs | 4 | 0% | | 12. | Spatial Awareness via Computer Vision | Object and scene awareness using webcam + ML | 3 | 0% | | 13. | Personalization Engine | Custom behavior based on user preferences and patterns | 3 | 0% | | 14. | Unified Visual Dashboard (Optional) | GUI showing command logs, AI responses, and image output | 4 | 0% | | 15. | Testing & Optimization | Unit + integration tests, latency and resource optimization | 3 | 0% | | 16. | Deployment & Documentation | Packaging the final release with setup guides | 2 | 0% | |

|  |
| --- |
| **6. Executed Work** |
| Certainly! Below is a **detailed "Executed Work" section** (more than 500 words) for **Progress Report 1** of an AI assistant project. This is suitable for academic submissions, internships, or formal documentation. **Executed Work – Progress Report 1**  1. **Project Initialization**  * Initialized project environment using Python 3.10.10 and created a structured codebase. * Configured .env files for securely storing API keys and credentials. * Installed essential packages including speechrecognition, pyttsx3, requests, and pyautogui. * Set up Git version control and pushed initial commits to GitHub repository. * Defined folder structure with separate modules for voice, APIs, UI, and utilities.  1. **Wake Word Detection**  * Developed a custom listener to detect the wake word “Hey FRIDAY” using continuous background monitoring. * Added threshold and keyword spotting logic to minimize false triggers. * Integrated startup and confirmation sound cues for better UX. * Optimized resource usage during idle listening mode.  1. **Voice Command Parsing**  * Implemented microphone input handling with Google Speech-to-Text API for accurate transcription. * Created intent parser to recognize command types like app control, search, and responses. * Added error handling for incomplete or noisy voice input. * Debug logging enabled for tracking parsed commands in development mode.  1. **Grok API Integration**  * Successfully connected Grok-1.5V for advanced multimodal AI interactions. * Enabled conversational capabilities for follow-up and reasoning queries. * Configured endpoints for both text-based and contextual image prompts. * Ensured asynchronous calls to maintain low-latency response times.  1. **Google API Integration**  * Integrated Google APIs for real-time web search, weather updates, and map lookups. * Structured output into short, voice-readable responses. * Built query constructor to convert natural language to actionable API requests. * Implemented fallback logic if API quota or connectivity fails.  1. **Text-to-Speech (TTS) System**  * Integrated pyttsx3 for text-to-speech response generation with adjustable rate and volume. * Built fallback mechanism for silent mode or TTS errors. * Added voice modulation and emotion tags for natural-sounding delivery. * Ensured compatibility with Windows voice engines.  1. **Desktop & Application Control**  * Implemented system control using os and pyautogui for actions like opening apps and managing windows. * Mapped voice commands like “Open Chrome”, “Close Notepad”, and “Minimize Window” to scripts. * Configured customizable command sets using JSON configuration. * Delivered voice feedback after each successful execution.  1. **Image Generation Module**  * Connected to HuggingFace APIs to generate images from text prompts. * Enabled voice-triggered image requests like “Show me a futuristic city at sunset.” * Implemented asynchronous image fetching and file-saving support. * Displayed outputs with optional preview or path-based access.  1. **Noise Filtering & Audio Processing**  * Applied input thresholding and silence detection to improve accuracy. * Calibrated microphone sensitivity across devices. * Filtered out background static and minimized misfires. * Enabled toggling between raw and filtered input for testing.  1. **Routing Engine and Dispatcher**  * Developed centralized command dispatcher to route parsed intents to appropriate modules (e.g., app control, TTS, APIs). * Modularized command handling for easy expansion of future features. * Added support for chaining responses (e.g., “What’s the time, then open YouTube”). * Logged all executed paths for debug and analytics purposes. |

# 

|  |
| --- |
| **7. Remaining Work** |
| As the project continues to progress, several core features and advanced functionalities remain to be developed and refined to ensure a fully intelligent, responsive, and personalized AI assistant. The upcoming development phases will focus on enhancing FRIDAY’s contextual understanding, expanding smart automation capabilities, and finalizing modules that enable seamless interaction across diverse environments.   * **Complete the Contextual Memory Engine** to allow FRIDAY to retain and recall past interactions, user preferences, and follow-up queries across sessions. * **Develop Smart Home Integration** to enable control of IoT-enabled devices (e.g., lights, thermostats, media systems) using voice commands through compatible APIs. * **Implement Spatial Awareness Functionality** using webcam and computer vision techniques to detect physical surroundings, enabling context-aware actions and responses. * **Build the Personalization Layer** that adapts FRIDAY’s behavior, tone, and recommendations based on individual user habits, preferences, and feedback. * **Design a Unified Visual Dashboard (Optional)** that displays AI responses, recent commands, generated images, and real-time system status in a sleek, intuitive GUI. * **Add Advanced Multi-API Command Handling**, including command chaining, conditional logic (e.g., “If it’s raining, remind me to carry an umbrella”), and response customization. * **Optimize Performance** by reducing latency in API calls, memory usage, and improving concurrency for simultaneous voice and API processing. * **Conduct Full-Scale Testing and Debugging**, including unit tests, integration tests, and real-world user simulations to ensure system robustness and reliability. * **Prepare Complete Documentation and Deployment Scripts** to support local and cloud-based deployment, along with user onboarding guides. * **Explore Additional Capabilities**, such as emotion-based response generation, integration with calendar/reminder tools, multi-language support, and offline command handling. |

|  |
| --- |
| **8. Reference** |
| *1. A. Vaswani et al., Attention Is All You Need, NeurIPS, 2017.*  *2. J. Brownlee, Deep Learning for Natural Language Processing, Machine Learning Mastery, 2018.*  *3. T. Wolf et al., Transformers: State-of-the-Art Natural Language Processing, Hugging Face, 2020.*  *4. SpeechRecognition Library Documentation, [Online]. Available:* [*https://pypi.org/project/SpeechRecognition/*](https://pypi.org/project/SpeechRecognition/)*. [Accessed: 15-April-2025].*  *5. pyttsx3 – Text to Speech conversion library in Python, [Online]. Available:* [*https://pypi.org/project/pyttsx3/*](https://pypi.org/project/pyttsx3/)*. [Accessed: 15-April-2025].*  *6. Google Cloud, Speech-to-Text API Documentation, [Online]. Available: https://cloud.google.com/speech-to-text/docs. [Accessed: 16-April-2025].*  *7. Hugging Face, Transformers Documentation, [Online]. Available: https://huggingface.co/docs/transformers. [Accessed: 17-April-2025].*  *8. MDN Web Docs, Web Speech API – SpeechRecognition and SpeechSynthesis, [Online]. Available:* [*https://developer.mozilla.org/en-US/docs/Web/API/Web\_Speech\_API*](https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API)*. [Accessed: 18-April-2025].*  *9. Grok AI, xAI's Grok API Overview, [Online]. Available:* [*https://x.ai*](https://x.ai)*. [Accessed: 19-April-2025].*  *10. Vite, Next Generation Frontend Tooling, [Online]. Available:* [*https://vitejs.dev/guide/*](https://vitejs.dev/guide/)*. [Accessed: 20-April-2025].* |

**Evaluation by Project Guide(s) and Coordinator(s)**

Please ✓ if work is satisfactory or 🗴 if work is not satisfactory and therefore requires a revision.

|  |  |  |
| --- | --- | --- |
| **Section** | **Remarks by Project Coordinator(s)** | **Remarks by Project Guide** |
| 1. Project identification |  |  |
| 2. Project insights |  |  |
| 3. Relevant study material |  |  |
| 4. Objective/Scope |  |  |
| 5. Expected outputs |  |  |
| 6. Utilization |  |  |
| 7. Literature study/Data collection |  |  |
| 8. Methodology |  |  |
| 9. Executed work |  |  |
| 10. Remaining work |  |  |
| 11. References |  |  |
| 12. Gantt chart |  |  |
| **Overall performance** |  |  |
| ***Signature and date*** |  |  |