**Software Requirements Specification (SRS) for "Hawai'i Concierge Bot"**

**1. Introduction**

**1.1 Purpose**

The "Hawai'i Concierge Bot" is a web application designed to provide users with a personalized virtual assistant experience for exploring and learning about Hawaii. The bot leverages Google Cloud Vertex AI's generative models to answer questions using only verified .gov sources.

**1.2 Scope**

The chatbot operates as an AI concierge tailored to Hawaii-related queries. It offers users:

* Accurate and concise answers sourced exclusively from .gov websites.
* A responsive and user-friendly chat interface.
* Features to improve accessibility, such as adjustable font size and mobile compatibility.

The bot aims to enhance tourism and local engagement by providing verified, trustworthy, and locale-specific information.

**1.3 Product perspective:**

* **User interface:** The UI has an button to open an close the chat window and appears as an box in the bottom right corner. It has buttons and a textfield

**1.4 Definitions**

* **Vertex AI**: Google's cloud service for deploying machine learning models.
* **.gov Websites**: Official government websites and subdomains serving as the sole data sources for this application.
* **Chat Interface**: A user-friendly graphical interface for exchanging messages with the bot.

**2. System Requirements**

**2.1 Functional Requirements**

* **User Interaction**:
  + Users can type messages in a text box and receive responses from the bot.
  + The bot only responds to Hawaii-specific questions using .gov data.
* **Response Generation**:
  + Bot responses are generated using Vertex AI's "gemini-1.5-pro-002" model.
  + The chatbot declines to answer questions not related to Hawaii or unsupported by .gov data sources.
* **Accessibility**:
  + The chat window allows users to adjust font size for readability.
  + Users can toggle the visibility of the chat window.

**2.2 Non-functional Requirements**

* **Performance**:
  + Response time for user queries must be under 5 seconds.
* **Scalability**:
  + The system should handle concurrent user interactions without performance degradation.
* **Security**:
  + All communications should be encrypted.
  + Data sourced from .gov websites should remain unaltered and verified.
* **Cross-Browser Compatibility**:
  + The application should work seamlessly on modern browsers (Chrome, Firefox, Safari, Edge).

**3. User Stories**

1. **As a user**, I want to type questions about Hawaii into a chatbot so that I can get accurate information about the islands.
   * *Acceptance Criteria*: The chatbot responds with information verified from .gov sources.
2. **As a user**, I want to adjust the chat window's font size so that I can improve text readability.
   * *Acceptance Criteria*: Buttons labeled "A▲" and "A▼" change the chat text size dynamically.
3. **As a user**, I want the chat window to minimize or close when not in use so that I can focus on other parts of the website.
   * *Acceptance Criteria*: Clicking the chatbot icon toggles the visibility of the chat window.
4. **As a user**, I want to know if the bot cannot answer a question so that I do not rely on incomplete or inaccurate responses.
   * *Acceptance Criteria*: The chatbot declines to answer questions outside its scope or unsupported by .gov data.
5. **As a user**, I want to see previous conversations so that I can refer to past responses.
   * *Acceptance Criteria*: The chat window displays chat history during the session.

**4. Design Constraints**

* The system must integrate with Google Cloud's Vertex AI generative models.
* The bot's interface must utilize HTML, CSS, and JavaScript for frontend functionality.
* The backend must be built using Python's Flask framework.

**5. System Design**

**5.1 Frontend Design**

* **HTML Structure**: Provides a responsive chat window with a toggleable visibility feature.
* **CSS Styling**: Includes background themes reflecting Hawaiian aesthetics and clear text formatting for accessibility.
* **JavaScript Functions**: Manages font size adjustments, message scrolling, and chat window toggling.

**5.2 Backend Design**

* **Flask Application**:
  + Routes for serving the chat interface (GET) and handling user messages (POST).
  + Chat history is maintained in memory for the session.
* **Vertex AI Integration**:
  + Configured to use the "gemini-1.5-pro-002" model with predefined safety and generation settings.

**6. Safety Considerations**

* Responses generated by the chatbot are filtered to avoid harmful or inappropriate content.
* Strict adherence to .gov sources ensures data authenticity and relevance.