**Solution Design**

**Related Actors** **-** Young Children

**Functional Requirements:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Source** | **Priority** |
| **F01** | Real-time Translation | System shall allow user to use application features in real-time via live video. | Trivial | MUST |
| **F02** | Hand/Gesture Identification | System shall identify the hand/gesture of the user. | Domain Analysis | MUST |
| **F03** | Gesture Instruction | System shall provide instruction image to the user on how to perform sign language gestures correctly. | Stakeholder Analysis | MUST |
| **F04** | Gesture/Hand Feedback | System shall output message to the user when no hand is visible. | Developer Analysis | MUST |
| **F05** | Sign Language Recognition for Letters | System shall recognize sign language gestures for letters performed by the user. | Domain Analysis | MUST |
| **F06** | Sign Language Recognition for Basic Phrases | System shall recognize sign language gestures for basic phrases performed by the user. | Domain Analysis | MUST |
| **F07** | On-Screen Translation | System shall provide an on-screen overlay translation of the gestures to the user. | Stakeholder Analysis | MUST |
| **F08** | Subtitles Translation | System shall provide translated subtitles for gestures performed by user. | Stakeholder Analysis | MUST |
| **F09** | Translated Text Customization | System shall allow users to customize elements of translated text such color, size, etc. | Developer Analysis | MUST |
| **F10** | Startup Window | System shall display a startup page when the application is launched. | Trivial | MUST |
| **F11** | Start Button | System shall display a start button on the startup page to initiate the translator. | Trivial | MUST |
| **F12** | Exit Button | System shall display an exit button on the startup page to exit the application. | Trivial | MUST |
| **F13** | Settings Button | System shall display a settings button on the startup page to customize translated text. | Trivial | MUST |
| **F14** | Radio Buttons | System shall display radio buttons for On-Screen Display and Subtitles Text to allow users to select desired translation mode. | Trivial | MUST |

**Non-Functional Requirements:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Source** | **Priority** |
| **N01** | Security | System must be secure and protect user data and prevent unauthorized access to systems. | Stakeholder Analysis | MUST |
| **N02** | Performance | System must respond quickly to user input and recognize the signs quickly and accurately in under 1 second. | Competitor Analysis | MUST |
| **N03** | Usability | System must provide a simple and intuitive interface that is user-friendly. | Domain Analysis | MUST |
| **N04** | Adaptability | System must be able to function in different lighting levels. | Developer Analysis | MUST |
| **N05** | Compatibility | System must be compatible with most Windows and Mac computers. | Stakeholder Analysis | MUST |
| **N06** | Security | System must adhere to user privacy laws and policies. | Stakeholder Analysis | MUST |
| **N07** | Responsiveness | System UI must be fluid and responsive. | Developer Analysis | SHOULD |
| **N08** | Error Handling | System must handle errors and provide clear error message to the user. | Developer Analysis | SHOULD |
| **N09** | Extensibility | System should be designed to accommodate additional features and new sign language data in the future. | Domain Analysis | SHOULD |
| **N10** | Modern User Interface | System shall have a modern user-friendly interface with appropriate elements for best user experience. | Stakeholder Analysis | SHOULD |

**Hardware Interface Requirements:**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Description** |
| **H01** | CPU | **Minimum** - Intel Core i3 | AMD Ryzen 3 | Apple M1 |
| **H02** | RAM | 4GB RAM or more |
| **H03** | HDD/SSD | At least 200MB for video analysis |
| **H04** | GPU | **Recommended** - Intel HD Graphics 520 or more | NVIDIA GeForce GT 635 | Radeon HD 8470D |

**Software Interface Requirements:**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Description** |
| **S01** | Operating System (OS) | **Minimum** - Windows 7 | MacOS Yosemite |
| **S02** | Communication app | Visual Studio Code (**Recommended**) |
| **S03** | Programming Language | Python |
| **S04** | Libraries Used | numpy, pandas, keras, Scikit-Image, customtkinter, cvzone, … |

**Solution Diagram:**

Diagram

Description automatically generated

**Solution Description:**

**Introduction:**

SignPal is a python application that allows users to make sign language gestures and receive real-time feedback. The system has a user-friendly interface and supports a variety of features such as hand/gesture identification, gesture instruction, sign language recognition, on-screen translation, and subtitle translation. The system complies with user privacy laws and policies and is designed to accommodate new features and sign language data in the future.

**Solution Components:**

**Real-time Translation:** This utilizes a video capture device, such as a webcam, that will record the user's movements and feed them into the system.

**Hand/Gesture Identification:** This can be achieved by incorporating a computer vision package known as *CVZone* that will analyze the video frames and identify the position and orientation of the user's hand as well as the gestures that are being performed.

**Gesture Instruction:** Before the translation process starts, an instruction image is displayed to the user on how to position and perform the sign language gestures correctly.

**Gesture/Hand Feedback:** This will be done using exception handling that will detect when the user's hand is not visible and display an appropriate error message.

**Sign Language Recognition (Letters and Basic Phrases):** This will be done by passing the frames of the live video into a *Trained Model* that recognizes the sign language gestures used.

**On-Screen/Subtitles Translation:** The *Trained Model* will output a translation when hand gestures frames are passed through it. The translation display format will be based on the choice the user made at the start i.e., On-Screen Overlay and Text Subtitles.

**Startup Window:** A startup window that will appear upon running the application. It includes essential elements such as Radio Buttons, Start, Exit, and Settings Buttons. These elements are designed to provide optimal functionality for the application.

**Solution Process:**

Upon running SignPal, a user-friendly interface window appears containing various interactive elements. These include two *radio buttons* named "**On-Screen Overlay**" and "**Text Subtitles**", as well as *buttons* for "**Start**", "**Settings**", and "**Exit**". The user can select between the two radio buttons, and upon clicking the "**Start**" button, the system initializes the sign language translation process.

Before proceeding with the translation, the user is presented with an instructional image that displays the optimal hand placement required for accurate interpretation. Once the user confirms their readiness by clicking the "**OK**" button, a live video feed is displayed, and the user can begin executing sign gestures.

The system captures and processes the user's sign gestures in real-time while simultaneously displaying their corresponding translations. Depending on the user's selected preference, these translations can appear either as an **On-Screen Overlay** which displays the translated word above the location of the user's hand in the live video feed, or as **Subtitles Text** which appears at the bottom of the screen.

Moreover, the system allows the user to customize the text size and colour of the displayed translations by clicking the "**Settings**" button in the startup menu. Overall, SignPal provides a seamless and user-friendly solution for real-time sign language translation.

**Real-World Example:**

SignPal could be used by a deaf person to communicate with someone who does not know sign language. For example, a deaf individual might use the system to translate their sign language gestures into text or speech in order to communicate with a doctor, lawyer, or other professional who does not know sign language.

In such a scenario, the deaf individual could use the system to translate their signs into spoken language or text, which could be displayed on a screen or read aloud by the system. This would enable the non-sign language speaker to understand and respond to the deaf individual's communication, allowing for more effective communication and potentially better outcomes.

Another potential use case might be in social situations, such as when a deaf individual is trying to meet and communicate with new people who do not know sign language. In this case, the system could provide a means for the deaf individual to communicate more effectively and engage in conversation with others, potentially reducing social isolation and improving quality of life.

SignPal has the potential to improve communication and facilitate social interactions for deaf individuals, even in situations where others do not know sign language.

**Machine Learning Algorithms:**

…

**Data Description:**

**Introduction:**

To ensure that SignPal recognizes and translates the hand signs with utmost accuracy, the model used for it would need to be trained, tested, and validated using a dataset of images with their corresponding labels.

**Dataset Description:**

The dataset should contain images of a hand performing different sign language gestures along with its respective translation (labels) in English. Each hand sign should have multiple versions of it such as rotated, flipped, zoomed in, and so on. It is important that the dataset is diverse and includes various sign languages and variations of sign language gestures to ensure the application is effective for a broad range of users.

For the effective training, testing, and validation of SignPal, a diverse and comprehensive dataset of sign language gestures should be used. This dataset should consist of images of a hand performing different sign language gestures, accompanied by its corresponding English translations (labels). It is essential that each hand sign has multiple versions (such as rotated, flipped, and zoomed in) to capture the variability that may be encountered in real-world scenarios.

Finally, to ensure the dataset's quality, it should be balanced and contain an equal number of samples for each gesture and each variation, to prevent bias towards certain signs or versions.

**Splitting the Dataset:**

The dataset will be split into **training** set, **validation** set, and **test** set. Initially, the ratio for it will be **80:10:10** respectively. With further progression, this ratio can be modified for better optimization of the model.

* The **training** set will be used to train the sign language model by inputting the hand sign images and its corresponding translated labels.
* The **validation** set will be used to fine-tune and optimize the model’s hyperparameters.
* The **test** set will be used to evaluate the model’s performance and accuracy after it has been fully trained and optimized. This set will not be used during the training or validation process to prevent bias in evaluation.

**Solution Motivation:**

The reasoning behind the proposed solution is as follows:

* The use of a video capturing device for real-time translation is a crucial feature as it is the fastest and the most effective method of getting input from the user. Moreover, providing translation instantaneously can be beneficial for the users in real-world situations, and can be a desirable and streamlined feature of the application.
* The **CVZone** computer vision package can be used for efficient hand/gesture recognition, as it provides swift hand detection and feedback. This eliminates the need to develop hand detection algorithms from scratch, thereby conserving time and resources that can be directed towards other mission-critical activities.
* Providing gesture feedback and customizable text settings can improve user experience and ensure that the system is accessible to a broader range of users.
* Displaying an instructional image before the translation process is an essential feature to ensure that the system can accurately interpret the hand gestures.
* The system’s ability to output translations and display them in either On-Screen Overlay or Text Subtitles provides flexibility and ease of use.
* Similarly, the startup window (UI) can also improve user experience while encouraging user interaction, user productivity, and assuring value for time.