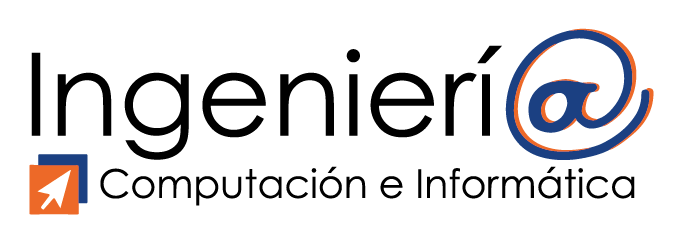
**UNIVERSIDAD DE TARAPACÁ**



**FACULTAD DE INGENIERÍA**

Departamento de Ingeniería en Computación e Informática



**Informe Final  
Escanear y Leer**

|  |  |
| --- | --- |
| **Autor(es):** | **Angel Alarcón**  **Fabián Justo**  **Mathiu Orellana**  **Raiza Ossandón** |
| **Asignatura:** | **Proyecto 2** |
| **Profesor(es):** | **Diego Aracena Pizarro** |
| **Grupo:** | **I** |

Arica, 08 de diciembre 2022

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor(es)** |
| 06/09/2022 | 1.0 | Inicio informe parte 1 con el formato requerido VBC | Fabián Justo |
| 07/09/2022 | 1.1 | Actualización del informe  Organización del proyecto | Fabián Justo  Mathiu Orellana |
| 13/09/2022 | 1.2 | Resumen del proyecto  Organización del proyecto  Planificación procesos de gestión | Mathiu Orellana  Raiza Ossandón  Angel Alarcón  Fabian Justo |
| 14/09/2022 | 1.3 | Versión preliminar del informe completo | Mathiu Orellana  Raiza Ossandón  Fabian Justo  Angel Alarcón |
| 15/09/2022 | 1.4 | Revisión de referencias | Mathiu Orellana |
| 11/10/2022 | 1.5 | Actualización del objetivo del proyecto | Mathiu Orellana |
| 12/10/2022 | 2.0 | Corrección de informe  y Casos de uso | Mathiu Orellana  Fabian Justo |
| 13/10/2022 | 2.1 | Actualización de escenarios  y Caso de uso | Mathiu Orellana |
| 14/10/2022 | 2.2 | Finalización Informe Avance 2 | Mathiu Orellana  Fabian Justo |
| 06/12/2022 | 3.0 | Implementación de los 2 informes corregidos  Implementación de manual de usuario | Angel Alarcón  Mathiu Orellana |
| 09/12/2022 | 3.1 | Implementación del funcionamiento del proyecto | Angel Alarcón  Fabian Justo |

## Historial de versiones

# Índice

[Historial de versiones 2](#_Toc121520526)

[Índice 3](#_Toc121520527)

[Panorama General 4](#_Toc121520528)

[Resumen del Proyecto 4](#_Toc121520529)

[Problema 4](#_Toc121520530)

[Solución 4](#_Toc121520531)

[Propósito 4](#_Toc121520532)

[Alcance 5](#_Toc121520533)

[Organización del Proyecto 7](#_Toc121520534)

[Personal y entidades internas 7](#_Toc121520535)

[Roles y responsabilidades 7](#_Toc121520536)

[Mecanismos de Comunicación 7](#_Toc121520537)

[Planificación de los procesos de gestión 8](#_Toc121520538)

[Lista de actividades 9](#_Toc121520539)

[Planificación de la gestión de riesgos 10](#_Toc121520540)

[Modelos de diseño 11](#_Toc121520541)

[Caso de Uso “Iniciar Programa” 12](#_Toc121520542)

[Diagrama de secuencia “Iniciar Programa” 13](#_Toc121520543)

[Caso de Uso “Analizar Texto” 14](#_Toc121520544)

[Diagrama de secuencia “Analizar texto” 15](#_Toc121520545)

[Caso de Uso “Reproducir Anterior” 16](#_Toc121520546)

[Diagrama de secuencia “Reproducir Anterior” 17](#_Toc121520547)

[Especificaciones 18](#_Toc121520548)

[Manual de instalación 19](#_Toc121520549)

[Instalación del programa 19](#_Toc121520550)

[Guía de uso 20](#_Toc121520551)

[Conclusión 22](#_Toc121520552)

[Referencias 23](#_Toc121520553)

# Panorama General

## Resumen del Proyecto

El proyecto LJR (Leer, Jugar, Reconocer) consiste en un software para dispositivos móviles, el cual facilita la lectura para la gente invidente utilizando reconocimiento óptico de caracteres (OCR) mediante una imagen y la traducción a voz mediante Text-To-Speech (TTS). El objetivo principal del software es lograr la accesibilidad para la gente invidente a textos y escritos, (principalmente escritos en imprenta), que no se puedan leer debido a la ausencia de una versión redactada en braille.

El software está restringido a una aplicación móvil para sistemas operativos Android con interfaz gráfica casi nula (debido a que esta será interactiva mediante voz), la cual vendrá acompañada con un soporte que facilitará la toma de imágenes del escrito.

## Problema

Las personas con discapacidad visual no son capaces de leer textos como lo hace la gente con capacidades de visión normales, debido a esto se han creado soluciones como el braille un sistema de escritura y lectura táctil, sin embargo, esta solución requiere de medios físicos para ayudar a la persona discapacitada a leer, además de ser una solución costosa. Es por esto que se desea establecer una solución que no requiera de medios físicos para la resolución del problema de la lectura para la gente con discapacidad visual.

## Solución

Desarrollar un software que a través de una cámara capta una imagen para procesarla por OCR, que da como resultado un texto el cual se leerá en voz alta.

## Propósito

El propósito del proyecto Escanear y Leer es ayudar al sector de la población invidente a leer textos escritos en imprenta y jugar juegos de cartas haciendo uso de tecnologías tales como OCR (reconocimiento óptico de caracteres), OpenCV (Visión computarizada) y TTS (texto a voz).

## Alcance

El proyecto comprenderá el desarrollo de una aplicación para el dispositivo Raspberry PI 3, la interfaz de la aplicación deberá ser accedida por medio de voz. La aplicación deberá tener un modo para la detección de textos.

**Objetivos**

* **Objetivo General**

Desarrollar un software capaz de reconocer oraciones escritas (principalmente en imprenta para posteriormente reproducir su lectura a través de un parlante.

* **Objetivos Específicos**
  + Estudiar las tecnologías a utilizar en la aplicación (OCR y OpenCV).
  + Estudiar Los frameworks que permitan utilizar cámaras, OCR y voz artificial
  + Exportar la aplicación para su uso en dispositivos móviles (principalmente Android).

**Restricciones**

* La interfaz debe ser accedida por medio de voz.
* La aplicación debe ser desarrollada para un dispositivo Raspberry PI.

**Entregables**

* Bitácoras (semanales)
* Informes parciales
* Presentaciones.
* Informe final.

**Escenarios:**

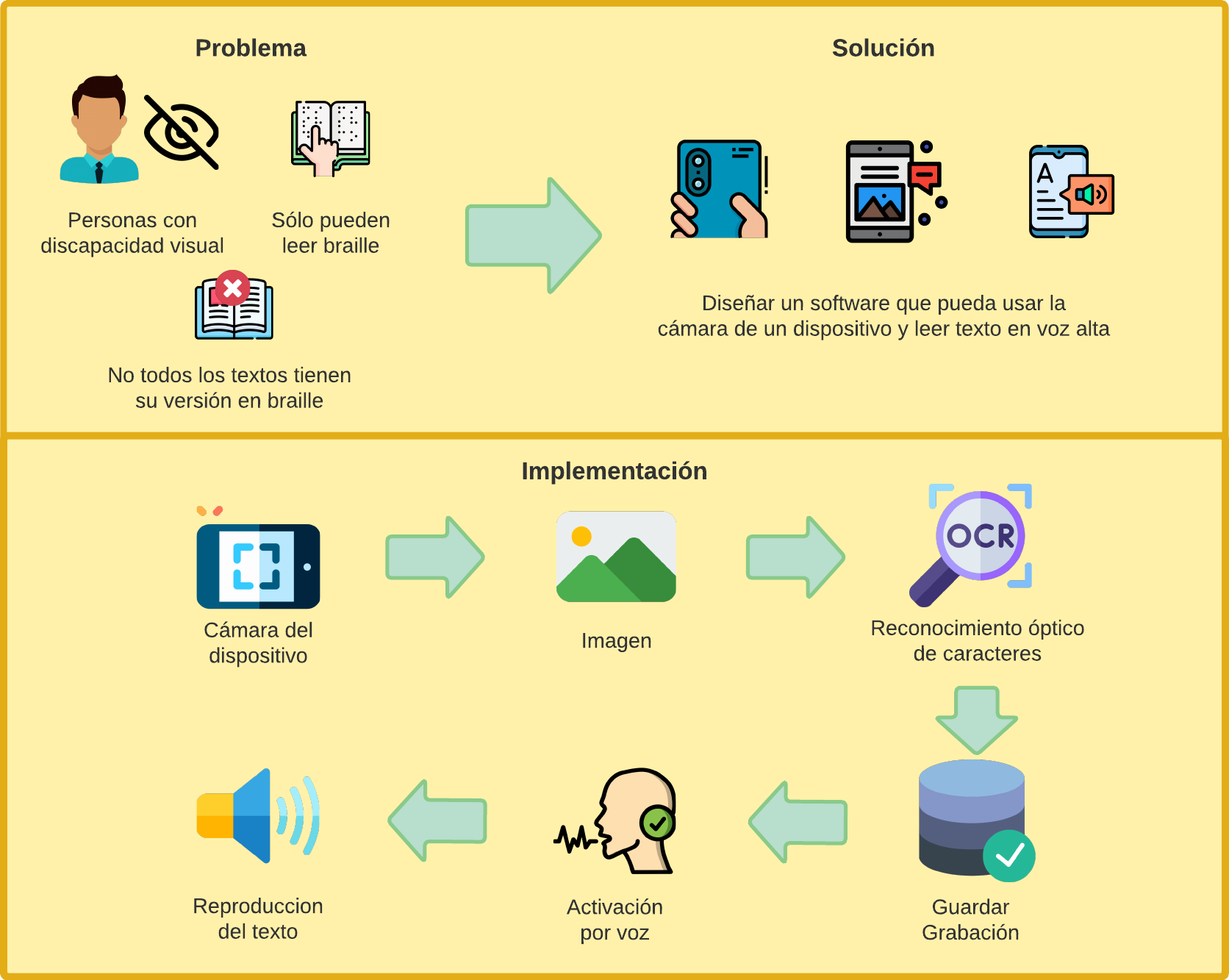
El siguiente escenario responde al problema de la lectura de textos no escritos en braille

Imagen n°1 “Escenario del problema de lectura de textos”

## Justificación de la arquitectura AADV

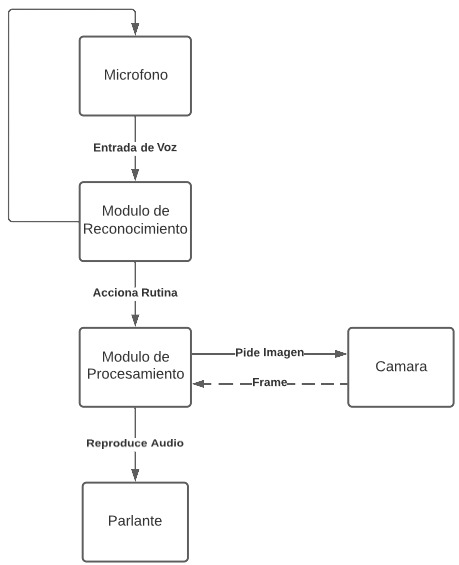


Imagen n°2 “Justificación de la arquitectura AADV”

# Organización del Proyecto

## Personal y entidades internas

* Jefe de Proyecto
* Diseñador Gráfico
* Encargado de documentación
* Encargado de Redmine
* Programador

## Roles y responsabilidades

* Jefe de Proyecto: Encargado de liderar el proyecto.
* Diseñador Gráfico: Encargado de diseñar las diferentes características visuales del proyecto (botones, iconos, interfaces visuales, etc).
* Encargado de Documentación: Encargado de documentar las diferentes características que se integrarán en el proyecto a través del transcurso del tiempo.
* Encargado de Redmine: Encargado de mantener la plataforma de redmine actualizada (Gantt, peticiones y wiki).
* Programador: Encargado(s) de escribir el código (según el lenguaje), perteneciente al programa del proyecto.
* Jefe programador: Encargado de revisar los códigos y dar el visto bueno a los mismos.

## Mecanismos de Comunicación

* Discord: Servicio de mensajería instantánea por medio de chat de texto y voz para la organización del grupo de trabajo fuera del área de clases.
* Redmine: Plataforma usada para el almacenamiento de la documentación y entregables del proyecto.
* Google Drive: Plataforma donde se alojan los documentos relacionados al proyecto (archivos ppt, docs, pdf), esta plataforma solo está disponible para miembros del proyecto.
* Google Docs: Plataforma para desarrollar los diferentes documentos y entregables del proyecto.

# Planificación de los procesos de gestión

**Estimación de Costos**

|  |  |
| --- | --- |
| **Software de desarrollo** | $0 |
| **Raspberry Pi 3** | $290.000 |
| **Notebook** | $600.000 |
| **Sueldo Desarrollador Junior** | $750.000 / mes |
| **Tiempo de Codificación** | 3 meses |
| **Costo total** | **$11.695.000** |

**Planificación de Recursos Humanos**

| **Diseñador** | Angel Alarcón |
| --- | --- |
| **Documentador** | Angel Alarcón |
| **Jefe Programador** | Fabián Justo |
| **Programador** | Raiza Ossandón  Fabián Justo  Mathiu Orellana  Angel Alarcón |
| **Jefe de proyecto** | Raiza Ossandón |
| **Encargado de Redmine** | Mathiu Orellana |

## Lista de actividades

Imagen n°3 “Carta Gantt”

## Planificación de la gestión de riesgos

|  |  |  |  |
| --- | --- | --- | --- |
| **RIESGOS** | **PROBABILIDAD DE OCURRENCIA** | **NIVEL DE IMPACTO** | **ACCIÓN REMEDIAL** |
| Problemas técnicos de código | 90% | 4 | -Verificar el código y arreglar el problema |
| Desgaste laboral | 60% | 3 | -Disminuir la carga de la persona afectada o apoyar en ella |
| Personal enfermo | 30% | 2 | - Aplazar el trabajo del integrante  -Distribuir el trabajo del integrante en el resto del equipo |
| Corte de electricidad | 20% | 3 | -Esperar el reintegro de electricidad  -Hacer uso de la batería del notebook |
| Pérdida del código base | 10% | 1 | -Tener un repositorio de respaldo para impedir este riesgo a toda costa |
| Malfunción o pérdida de Hardware | 10% | 1 | -Hacer uso del hardware proporcionado por la universidad |
| Malfunción de Software | 10% | 2 | -Reinstalar o buscar solución del error. |
| Pérdida de un integrante | 5% | 1 | -Dividir el trabajo que le tocaba a aquel integrante entre los que quedan |

## Modelos de diseño

**Diagrama de clases “Usuario”**

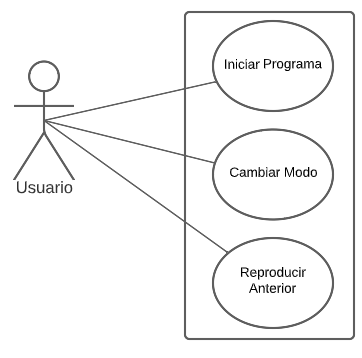
****

Imagen n°4 “Diagrama de clases de usuario”

**Diagrama de clases “Sistema”**

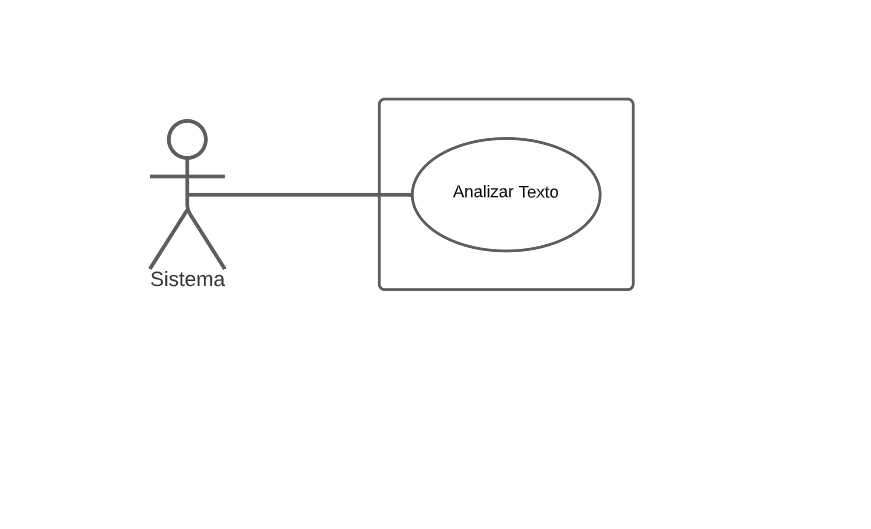
****

Imagen n°5 “Diagrama de clases del sistema”

## Caso de Uso “Iniciar Programa”

|  |  |
| --- | --- |
| **Nombre** | Iniciar Programa |
| **Descripción** | El programa inicia |
| **Condiciones previas** | - |
| **Actores** | Usuario |
| **Secuencia normal** | 1. El usuario inicia el programa. 2. El sistema automáticamente inicia el “modo Lectura”. 3. El sistema avisa al usuario que presente algo para escanear. |
| **Secuencia alternativa** |  |
| **Postcondición** | El sistema ha iniciado con su “modo lectura”. |

## 

## Diagrama de secuencia “Iniciar Programa”

Imagen n°6 “Diagrama de secuencias iniciar programa”

## Caso de Uso “Analizar Texto”

|  |  |
| --- | --- |
| **Nombre** | Analizar texto |
| **Descripción** | La cámara realiza una captura para ser analizadas por el sistema en búsqueda de texto |
| **Condiciones previas** | El sistema debe encontrarse en modo lectura |
| **Actores** | Cámara |
| **Secuencia normal** | 1. La cámara realiza constantes capturas 2. El sistema analiza las capturas en búsqueda de texto. 3. El sistema detecta texto y lo reproduce a través del parlante |
| **Secuencia Alternativa** | 1. Si el sistema no detecta texto o este es ilegible reproducirá un sonido en forma de alerta para el usuario. |
| **Poscondición** | El sistema almacena el último texto leído. |

## Diagrama de secuencia “Analizar texto”

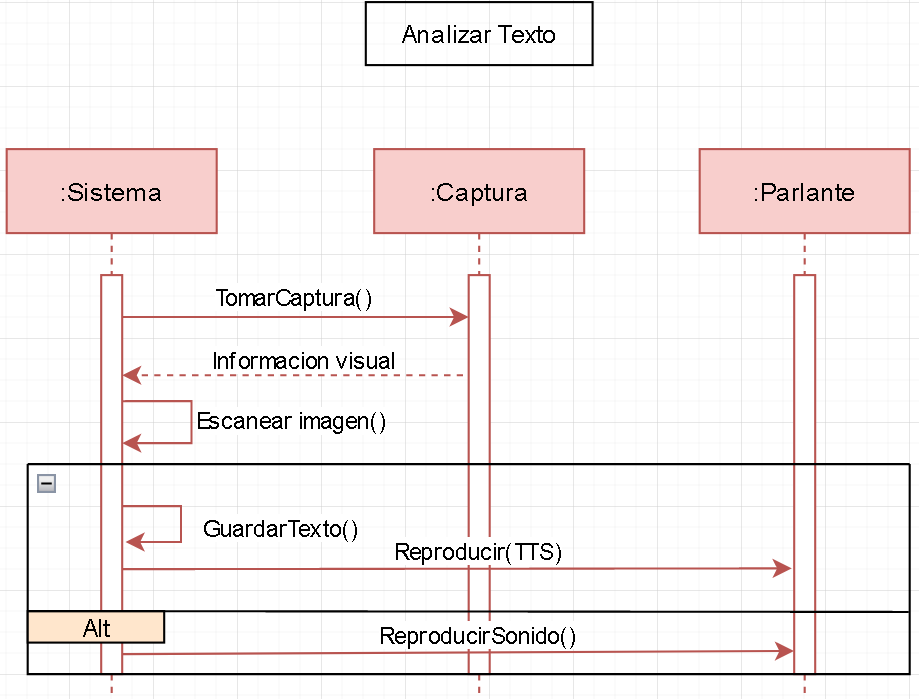
****

Imagen n°7 “Diagrama de secuencias Analizar texto”

## Caso de Uso “Reproducir Anterior”

|  |  |
| --- | --- |
| **Nombre** | Reproducir Anterior |
| **Descripción** | El sistema lee en voz alta el último texto o mencionado. |
| **Condiciones previas** | Debe existir un texto anteriormente almacenado. |
| **Actores** | Usuario |
| **Secuencia normal** | 1. El usuario dice “Repetir”. 2. El sistema accede a su archivo de texto. 3. El sistema reproduce mediante text to speech el texto a través del parlante. |
| **Secuencia alternativa** | 1. Si el sistema no tiene almacenado un texto previo, este le hará saber al usuario mediante voz. |
| **Poscondición** | - |

## Diagrama de secuencia “Reproducir Anterior”

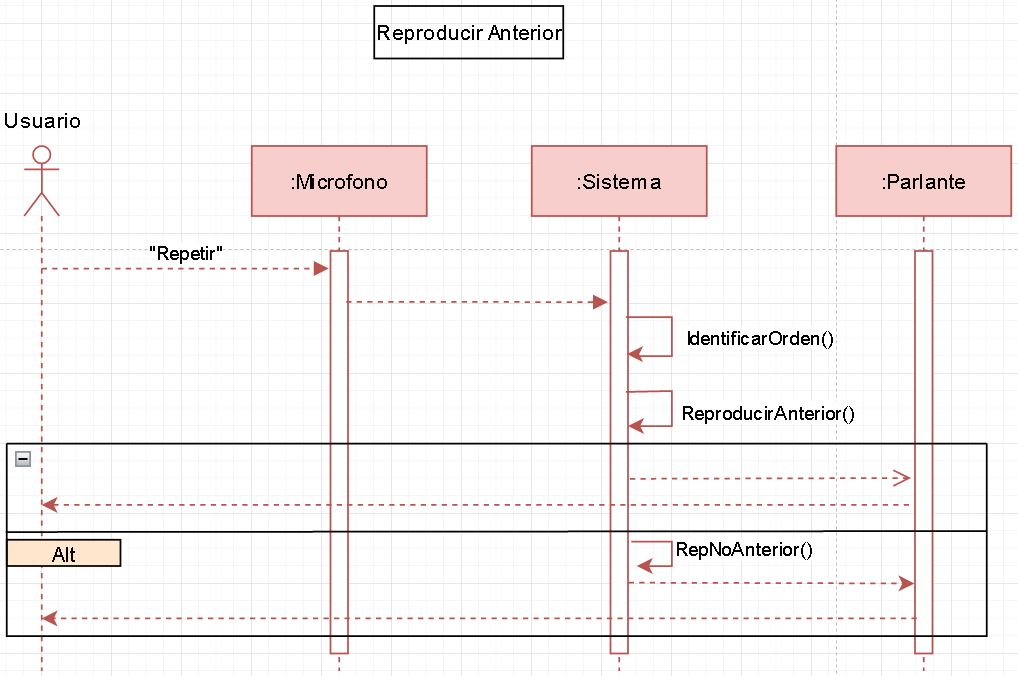


Imagen n°8 “Diagrama de secuencias Reproducir anterior”

# Especificaciones

1. La aplicación será desarrollada en Python.
2. Hará uso de las tecnologías OCR/Tesseract y OpenCV.
3. Tendrá una interfaz interactiva por medio de voz.
4. Deberá usar el siguiente hardware del dispositivo Raspberry pi 3: Micrófono, Cámara y altavoz.
5. Detectará oraciones principalmente en imprenta proveniente de imágenes tomadas por la cámara del dispositivo.
6. Leerá las oraciones detectadas en voz alta (text to speech).
7. Los comandos de voz serán: “Leer”,” Salir”, “Captura”,” Inicio”, “Repetir”.
8. Los principales lenguajes a reconocer serán: español e inglés.
9. La app tendrá dos idiomas para usar dentro de ella: español e inglés.

## Manual de instalación

Es de suma importancia conocer que el sistema no presenta interfaz gráfica ya que este fue desarrollado con la intención de que el usuario acceda a las funciones del mismo a través de órdenes de voz. Además, se recomienda instalar el programa en Raspberry Pi (Hardware para el cual se desarrolló el programa), sin embargo, el programa es capaz de ejecutarse en computadores convencionales.

Es indispensable tener una cámara, micrófono y parlante para el funcionamiento del programa.

Con este programa se tiene la intención de que personas con discapacidades visuales (usuario objetivo) puedan acceder a la información impresa.

## Instalación del programa

Para la ejecución del programa se requiere de los siguientes recursos hardware:

* Raspberry Pi (Recomendado) con los siguientes dispositivos/características:
  + Conexión a internet.
  + Cámara.
  + Micrófono.
  + Parlante.

También se requieren de los siguientes recursos software, cabe aclarar que las líneas de código para la correspondiente instalación de cada librería, se debe ingresar por la terminal de la Raspberry Pi:

* Python 3 (en adelante) con las siguientes librerías:
  + Pyaudio
    - sudo apt-get install python-pyaudio
  + Pytesseract
    - sudo apt-get install tesseract-ocr
  + SpeechRecognition
    - python3 -m speech\_recognition
  + OpenCV
    - sudo pip install opencv-contrib-python

## Guía de uso

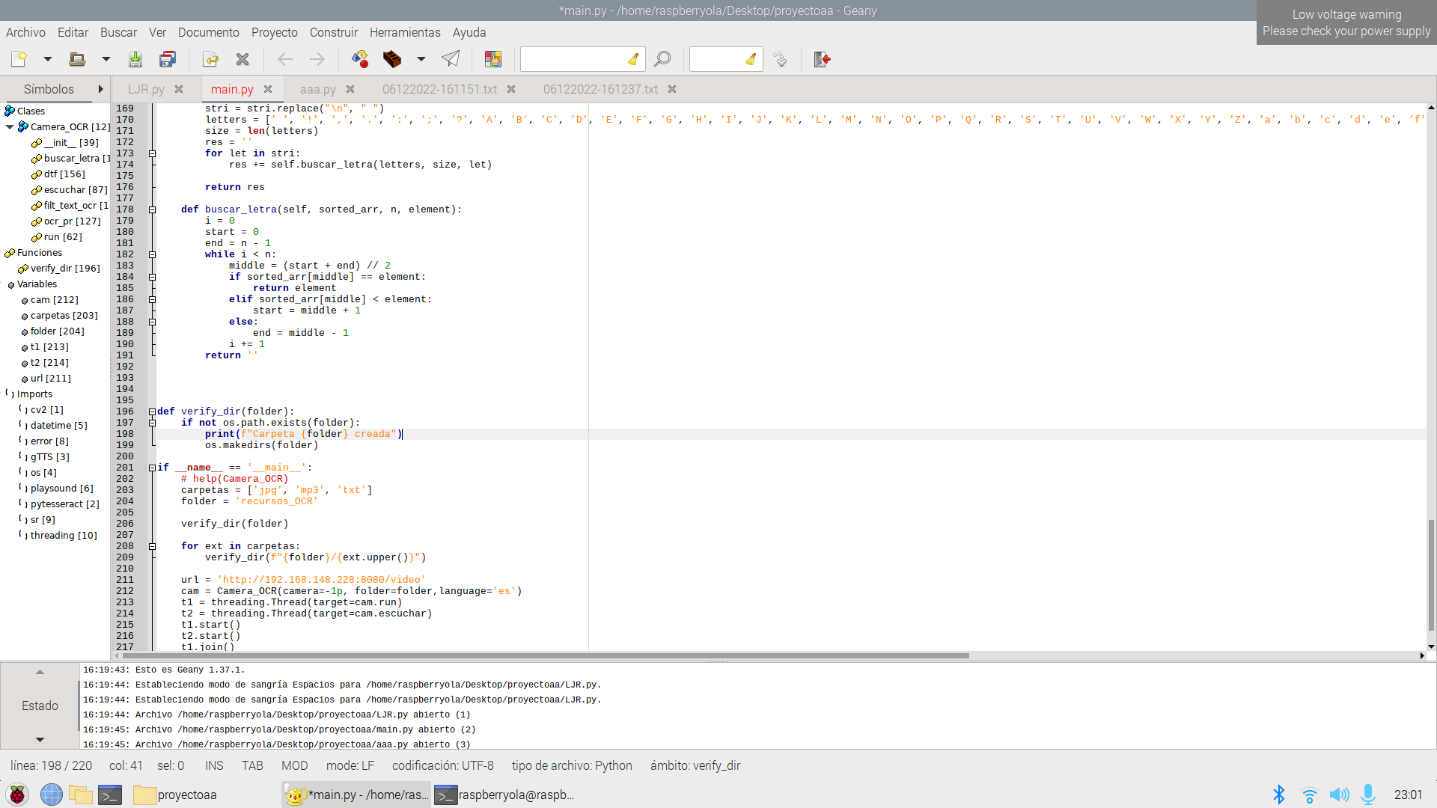


Imagen n°9 “Código del programa”

Una vez establecido el código en la aplicación, al ejecutarlo se abrirá una terminal, en donde el usuario tendrá que decir en voz alta “Leer”, al decir la frase se abrirá automáticamente la cámara, para que el usuario pueda poner el texto.

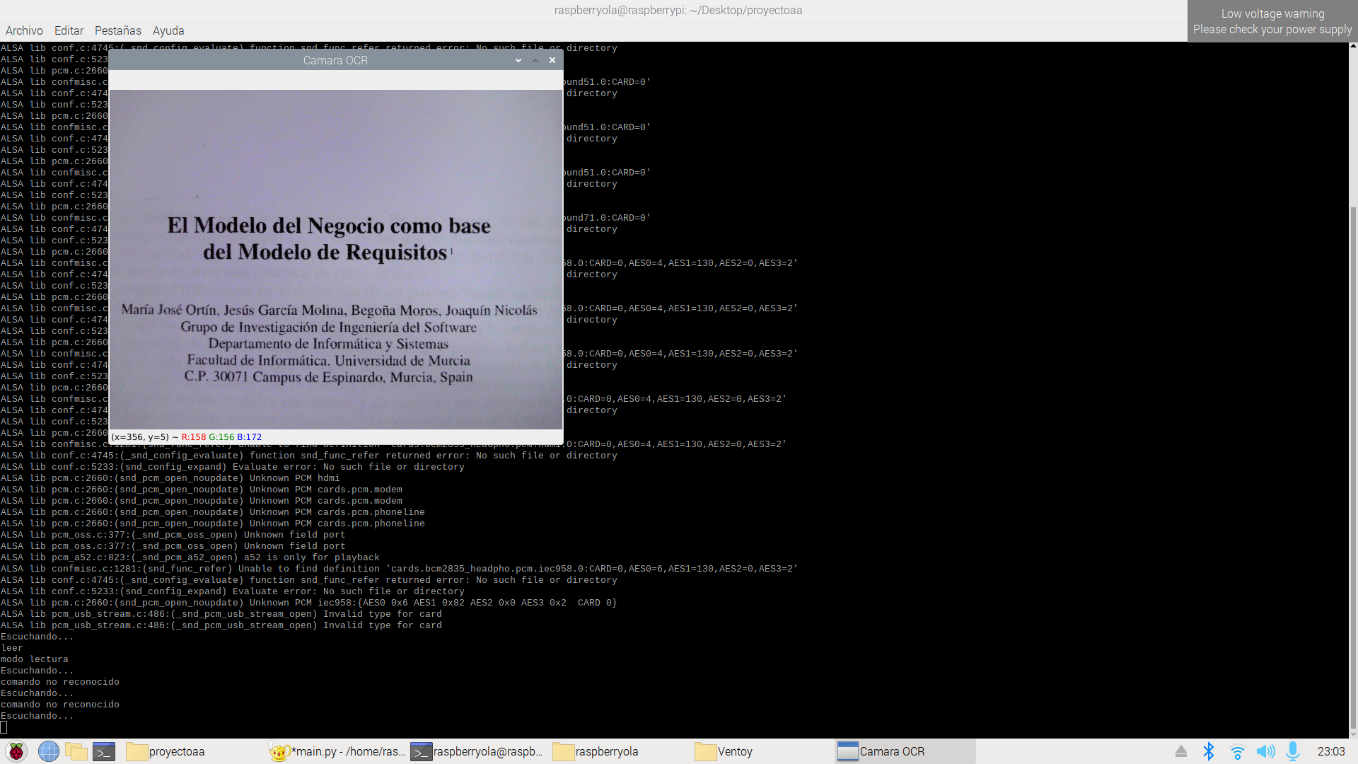


Imagen n°10 “Funcionamiento de la aplicación”

Al poner el texto en frente de la cámara, el usuario en voz alta dirá “Captura”, luego la cámara se congelará por unos segundos, esto significa que está procesando la foto y generando el archivo de audio, imagen y texto respectivo.

Una vez generados, se reproducirá automáticamente el archivo de audio respectivo y la cámara seguirá funcionando para que se pueda escanear otra foto nuevamente.

Para poder apagar la cámara, el usuario tendrá que decir “Inicio” y se cerrará la cámara automáticamente, y luego, para poder cerrar el programa, el usuario en voz alta tendrá que decir, “Salir” y se terminará el programa.

# Conclusión

El proyecto Escanear y Leer fue un proyecto que, a pesar de su simplicidad, su elaboración llevó contantes problemas y soluciones, además de cambios de planificación.

En base a todo el proyecto realizado, se estima y se sostiene que este proyecto es viable en términos económicos y de grado de dificultad. Sin embargo, también se da cuenta de varios puntos que antes no se tuvieron en cuenta y se debieron reforzar para concluirlo: primero, los casos de uso rescatados eran escasos y por tanto se debió dar más énfasis en este aspecto del proyecto, con tal de crear y mantener un sistema estable y seguro.

Segundo, a raíz de uno de los puntos mencionado, se descubrió que un escenario que se estimaba terminar por problemas de software y incompatibilidades se debió de cancelar, este escenario fue destinado con el tema del juego de cartas “Uno”. El cual dicho anteriormente, por problemas se eliminó de la entrega final.

De la mano de los problemas, uno principal fue el que afectó en los pilares fundamentales del proyecto en donde se mencionaba realizar esta aplicación para dispositivos móviles (Con el sistema operativo Android en específico), lo cual comenzó a generar diversos problemas casi al concluir el proyecto, problemas tales como al momento de exportar la aplicación al dispositivo, y al ejecutar la aplicación dentro del dispositivo, concluyendo en cambiar de ambiente y centrarse en el funcionamiento del proyecto en el dispositivo Raspberry pi 3.

# Referencias

* Sueldo promedio programador junior “<https://www.chiletrabajos.cl/sueldos/programador/junior>”
* Precio promedio notebook en chile “<https://www.df.cl/tendencias/tecnologia/cuanto-cuesta-un-computador-en-chile-los-paises-mas-caros-de-la-region>”
* Precio promedio de smartphones “<https://www.google.com/search?client=firefox-b-d&q=317+usd+to+clp>”
* Establecer conexión de cámara de celular con Python “<https://opencv.org>” “[https://www.geeksforgeeks.org/connect-your-android-phone-camera-to-opencv-python](https://www.geeksforgeeks.org/connect-your-android-phone-camera-to-opencv-python/)”
* Implementación de OCR en Python “<https://nanonets.com/blog/ocr-with-tesseract/>”