



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



## TFG del Grado en Ingeniería Informática

**Detección y evaluación de  
ejercicios de rehabilitación  
para pacientes con la  
enfermedad de Parkinson  
Documentación Técnica**



Presentado por Lucía Núñez Calvo  
en Universidad de Burgos — 7 de julio de 2022  
Tutor: José Francisco Díez Pastor, José Luis  
Garrido Labrador y José Miguel Ramírez Sanz



---

# Índice general

---

<b>Índice general</b>	i
<b>Índice de figuras</b>	iii
<b>Índice de tablas</b>	v
<b>Apéndice A Plan de Proyecto Software</b>	1
A.1. Introducción . . . . .	1
A.2. Planificación temporal . . . . .	2
A.3. Estudio de viabilidad . . . . .	9
<b>Apéndice B Especificación de Requisitos</b>	13
B.1. Introducción . . . . .	13
B.2. Objetivos generales . . . . .	13
B.3. Catalogo de requisitos . . . . .	14
B.4. Especificación de requisitos . . . . .	17
<b>Apéndice C Especificación de diseño</b>	27
C.1. Introducción . . . . .	27
C.2. Diseño de datos . . . . .	28
C.3. Diseño procedimental . . . . .	32
C.4. Diseño arquitectónico . . . . .	35
<b>Apéndice D Documentación técnica de programación</b>	37
D.1. Introducción . . . . .	37
D.2. Estructura de directorios . . . . .	37
D.3. Manual del programador . . . . .	41

D.4. Compilación, instalación y ejecución del proyecto . . . . .	44
D.5. Fallos y soluciones . . . . .	45
D.6. Pruebas del sistema . . . . .	50
<b>Apéndice E Documentación de usuario</b> . . . . .	<b>53</b>
E.1. Introducción . . . . .	53
E.2. Requisitos de usuarios . . . . .	53
E.3. Instalación . . . . .	53
E.4. Manual del usuario . . . . .	54
<b>Bibliografía</b> . . . . .	<b>61</b>

---

# Índice de figuras

---

B.1. Diagrama de casos de uso para el terapeuta. . . . .	17
B.2. Diagrama de casos de uso para el usuario. . . . .	18
C.1. Diagrama de clases del proyecto. . . . .	29
C.2. Diagrama de clases de la aplicación. . . . .	31
C.3. Diagrama de secuencias de la parte encargada de obtener las posiciones. . . . .	32
C.4. Diagrama de secuencias de la parte encargada de localizar las secuencias. . . . .	33
C.5. Diagrama de secuencias de la aplicación. . . . .	34
C.6. Diagrama de despliegue de las máquinas virtuales <i>docker</i> . . . . .	35
C.7. Diagrama de paquetes de la aplicación. . . . .	36
C.8. Diagrama de paquetes de la aplicación. . . . .	36
D.1. Error de tipo <i>Caused by: java.io.EOFException</i> . . . . .	46
D.2. Error de tipo <i>StreamingContext: StreamingContext has already been stop-ped</i> . . . . .	47
D.3. Error de tipo <i>The server is not running</i> . . . . .	47
D.4. Error de tipo <i>Get permission denied while trying to connect to the Docker daemon socket at unix</i> . . . . .	48
D.5. Error de tipo <i>kafka</i> . . . . .	49
E.1. Aplicación en modo ejemplo ON. . . . .	55
E.2. Aplicación en modo ejemplo OFF. . . . .	55
E.3. Carga y descarte de vídeos. . . . .	56
E.4. Recorte de vídeos en modo ejemplo ON. . . . .	59
E.5. Recorte de vídeos en modo ejemplo OFF. . . . .	60
E.6. Aviso de falta de frames. . . . .	60

E.7. Detección de animales usando diferentes algoritmos. . . . .	60
--	----

---

# Índice de tablas

---

A.1. Tareas del <i>sprint 1</i> . . . . .	2
A.2. Tareas del <i>sprint 2</i> . . . . .	3
A.3. Tareas del <i>sprint 3</i> . . . . .	3
A.4. Tareas del <i>sprint 4</i> . . . . .	4
A.5. Tareas del <i>sprint 5</i> . . . . .	5
A.6. Tareas del <i>sprint 6</i> . . . . .	6
A.7. Tareas del <i>sprint 7</i> . . . . .	7
A.8. Tareas del <i>sprint 8</i> . . . . .	8
A.9. Tareas del <i>sprint 9</i> . . . . .	8
A.10. Costes de personal. . . . .	9
A.11. Costes de <i>hardware</i> . . . . .	10
A.12. Costes de servicios. . . . .	10
A.13. Costes finales. . . . .	10
A.14. Tabla con las licencias de las librerías y herramientas utilizadas.	11
B.1. Caso de uso 0: Insertar vídeos. . . . .	19
B.2. Caso de uso 1: Localiza el vídeo. . . . .	20
B.3. Caso de uso 2: Reproducir vídeos. . . . .	21
B.4. Caso de uso 3: Cambiar modo. . . . .	22
B.5. Caso de uso 4: Seleccionar vídeos de la lista. . . . .	23
B.6. Caso de uso 5: Ver vídeo resultado. . . . .	24
B.7. Caso de uso 6: Introducir <i>frames</i> de inicio y final. . . . .	25



## *Apéndice A*

---

# **Plan de Proyecto Software**

---

## **A.1. Introducción**

La planificación de un proyecto es una fase fundamental para su correcto desarrollo. En este apartado se comentará, por una parte, la planificación temporal del proyecto mediante los distintos *sprints* que se han llevado a cabo, y por otra parte, se realizará un breve estudio sobre la viabilidad del proyecto.

En este proyecto se ha utilizado la metodología ágil *Scrum*, esta metodología tiene como objetivo base la creación de diferentes iteraciones o **sprints** [8]. Para una correcta planificación, se optó por realizar *sprints* de dos o tres semanas de duración. Aunque las reuniones se realizaban semanalmente, los objetivos se planteaban con un margen de varias semanas y en las reuniones posteriores se iban comentando las dudas y los avances.

1. **Planificación Temporal**, se expondrán las diferentes tareas realizadas en cada uno de los *sprint*, acompañadas del coste estimado que se propuso para ellas y el coste final que implicó su realización.
2. **Estudio de viabilidad**, en este apartado se comprobará si el proyecto realizado tiene una viabilidad legal y económica.

## A.2. Planificación temporal

La planificación temporal se ha dividido en distintos *sprints* de dos o tres semanas de duración cada uno, en los cuales se realizaron distintas tareas y reuniones. Las reuniones se realizaban semanalmente informando de las tareas realizadas y los problemas surgidos, a lo que aportaban distintas soluciones y tareas nuevas.

### Sprint 0: Noviembre - Enero

El *sprint 0* consistió en una primera toma de contacto con el proyecto que abarcó desde Noviembre del año 2021 a Enero del año 2022. En estos meses se realizaron reuniones periódicas cada dos semanas con el objetivo de comprender el proyecto de los alumnos José Luís Garrido Labrador y José Miguel Ramírez Sanz, pero sobretodo, asimilar y leer numerosos artículos sobre la metodología **DTW**(*Dynamic Time Warping*) y como puede implantarse a la búsqueda de secuencias y comparación o extracción de características de la pose humana.

### Sprint 1: 03/02/2022 - 24/02/2022

El *sprint 1* consistió en una investigación sobre diferentes librerías en *Python* que implementasen el algoritmo *DTW*. Además se descargó la plantilla aportada por la Universidad de Burgos para poder desarrollar la memoria en *LATEX*.

Tareas	Est.	Final
Investigación sobre paquetes DTW	10	13
Investigar sobre la librería tslearn	5	14
Investigar sobre la librería dtaidistance	5	3
Investigar sobre la librería pydtw	5	2
Investigar sobre la librería simuledtw	5	2
Investigar sobre la librería scipy	5	4
Descargar plantilla LATEX	1	1

Tabla A.1: Tareas del *sprint 1*.

La mayor dificultad en este *sprint* fue la comprensión de las diferentes librerías, con qué tipo de datos se podían utilizar y cómo analizar y comprender los resultados obtenidos.

## Sprint 2: 25/02/2022 - 10/03/2022

El *sprint 2* consistió en la realización de pruebas sobre los paquetes investigados y en la instalación de los programas necesarios para la ejecución del proyecto fin de máster, TFM-2020<sup>1</sup>.

Tareas	Est.	Final
Ejecutar múltiples pruebas sobre las librerías analizadas	5	7
Investigar como representar resultados visualmente	2	2
Crear un cuaderno con las pruebas realizadas	1	2
Comenzar con el equipo <i>gamma</i> <sup>2</sup>	2	4
Instalar <i>Nvidia, Cuda y Tensorflow</i>	2	4
Adaptar las versiones de los programas a las requeridas por el proyecto	2	7

Tabla A.2: Tareas del *sprint 2*.

La mayor dificultad de este *sprint* fue localizar las versiones compatibles entre los programas a instalar y conseguir instalarlo todo con éxito.

## Sprint 3: 11/03/2022 - 24/03/2022

El *sprint 3* consistió en la descarga y ejecución del proyecto *TFM-2020*. Además en este *sprint* se crearon una serie de vídeos cortos que más tarde serían utilizados en las pruebas del proyecto.

Tareas	Est.	Final
Descargar y probar el proyecto de partida	2	12
Crear vídeos cortos para poder probar el proyecto	1	1
Solucionar error de <i>Kafka</i>	2	2
Solucionar errores de compatibilidad	2	4
Solucionar errores de memoria	2	5
Documentar errores surgidos en las instalaciones	2	2
Documentar errores surgidos en las ejecuciones	2	3
Probar los paquetes <i>DTW</i> con datos de mayor tamaño unidimensionales	5	7

Tabla A.3: Tareas del *sprint 3*.

---

<sup>1</sup>Trabajo final de máster realizado por los alumnos José Luís Garrido Labrador y José Miguel Ramírez Sanz

La mayor dificultad de este *sprint* fue conseguir solucionar los errores de versiones que surgían al intentar ejecutar el proyecto. Finalmente se consiguió subsanar muchos errores y ejecutar el proyecto, aunque este de momento no estaba listo para ser usado ya que se consiguió ejecutar pero no se consiguieron los resultados esperados en cuanto a la extracción de *frames*.

### Sprint 4: 25/03/2022 - 07/04/2022

El *sprint 4* consistió en realizar numerosas pruebas sobre los algoritmos *DTW* encontrados pero, en este caso sobre conjuntos de datos de mayor tamaño. También se analizaron tanto los resultados arrojados como los tiempos de ejecución transcurridos. Por otra parte fue clave el análisis que se hizo sobre el funcionamiento interno de los contenedores para comprender los posibles errores.

Tareas	Est.	Final
Ejecutar pruebas con datos de mayor tamaño	6	8
Ejecutar pruebas con datos multidimensionales	7	10
Transferir archivos entre equipos usando el comando scp	1	1
Solucionar errores del proyecto	4	10
Analizar los contenedores Docker que se lanzaban en cada ejecución	3	5
Analizar las imágenes Docker	4	4
Ejecutar contenedores Docker por separado	3	3
Analizar los log de error	2	2
Redactar nuevos errores y sus soluciones	1	1

Tabla A.4: Tareas del *sprint 4*.

La mayor dificultad de este *sprint* fue familiarizarse con la tecnología *Docker*. Se tuvieron que revisar cada una de las imágenes y analizar los contenedores individualmente. Finalmente se logró comprender la razón de muchos errores y el funcionamiento interno del proyecto.

### Sprint 5: 08/04/2022 - 28/04/2022

El *sprint 5* consistió en la creación de nuevos conjuntos de datos, en concreto la creación de diversas series temporales para ejecutar múltiples pruebas sobre los algoritmos *DTW* en conjuntos de datos mucho más dispersos y de mayor tamaño.

Tras comprobar los resultados arrojados se procedió a implementar la búsqueda de secuencias con datos reales. Para ello se extrajo el inicio, el final y la secuencia intermedia de cada secuencia obtenida, con estas secuencias de mucho menor tamaño se empezaron a realizar las pruebas de localización de secuencias y posteriormente analizar los resultados arrojados.

Tareas	Est.	Final
Crear nuevos conjuntos de datos aleatorios, de mayor tamaño y con mayor ruido	1	1
Ejecutar múltiples pruebas sobre los nuevos conjuntos de datos	3	5
Crear nuevos conjuntos de datos similares a un patrón de referencia	1	1
Ejecutar múltiples pruebas en las que encuentre el conjunto de datos más parecido al patrón	3	5
Investigar métricas de comparación que informen como de parecidas son dos secuencias	2	4
Ejecutar múltiples pruebas sobre secuencias incompletas	3	3
Almacenar el punto de inicio y final en el que se encuentra una secuencia dentro de otra	1	1
Probar el funcionamiento de Detectron2 sobre cuadernos	3	3
Investigar múltiples algoritmos de detección de objetos	4	4
Ejecutar pruebas sobre los algoritmos encontrados	2	5
Ejecutar el proyecto sobre los distintos modos de ejecución	5	6

Tabla A.5: Tareas del *sprint 5*.

La mayor dificultad de este *sprint* fue la elección de algoritmos que obtuviesen buenos resultados en un tiempo óptimo. Muchos algoritmos obtenían unos buenos resultados con distintos tipos de datos pero sus ejecuciones podían superar los cuarenta o cincuenta minutos aproximadamente, en conjuntos de datos compuestos por  $K$  valores, siendo  $Z$  la longitud total del conjunto de datos, con un número entero comprendido entre  $Z \in [2500 : 3000]$  y  $NxM$  el tipo de datos que se almacena en cada valor de  $K$ , con dimensiones 27x2 o 27x3.

## Sprint 6: 29/04/2022 - 12/05/2022

El *sprint 6* consistió en la realización de nuevas pruebas sobre la localización de secuencias. Para ello, lo primero que se tuvo que realizar fue la

creación de nuevos vídeos y a partir de esos vídeos se obtuvieron las posiciones relativas al esqueleto del individuo. Tras realizar múltiples pruebas intentando localizar las secuencias obtenidas en otra secuencia de mayor tamaño y observar los resultados arrojados, se procedió a empezar con la realización de la memoria.

Tareas	Est.	Final
Creación de nuevos vídeos	1	1
Analizar código de <code>consumer.py</code>	2	2
Ejecuciones del programa sobre los nuevos vídeos	6	12
Analizar que sucede si en un frame no detecta el esqueleto	2	1
Investigar parámetros opcionales DTW	3	2
Investigar como almacenar todas las secuencias similares a un patrón de referencia	6	3
Crear un cuaderno con múltiples pruebas sobre como almacenar las secuencias	3	4
Analizar como afecta el Parkinson a los movimientos de las extremidades	3	3
Redactar la introducción de la memoria	1	1
Redactar un breve resumen para la memoria	1	2
Redactar <i>Conceptos teóricos</i>	2	1
Crear una estructura de directorios en GitHub	1	1

Tabla A.6: Tareas del *sprint 6*.

La mayor dificultad de este *sprint* fue la realización de las nuevas pruebas de localización de secuencias, ya que al tratarse de múltiples vídeos, se dedicó mucho tiempo a extraer todas y cada una de las posiciones y posteriormente a utilizarlas para la búsqueda de secuencias. Este trabajo resultó un poco tedioso y consumió demasiado tiempo ya que muchas veces las ejecuciones no terminaban favorablemente y había que repetir el proceso.

## Sprint 7: 13/05/2022 - 26/05/2022

El *sprint 7* consistió en modificar los ficheros ya existentes. Hasta este momento se habían obtenido los conjuntos de datos enteros y posteriormente se separaban según las posiciones que ocupasen. En este *sprint* el objetivo fue generar distintos ficheros con extensión `.pickle` según las posiciones relativas al esqueleto que contuviesen, es decir, poder separar entre posiciones de ángulos, posiciones del conjunto entero, únicamente posiciones sin ángulos, etc.

Otro de los objetivos de este *sprint* fue el de solucionaron nuevos errores surgidos en los contenedores *Docker* y en investigaron diferentes librerías para implementar una aplicación de escritorio.

Tareas	Est.	Final
Modificar el fichero <code>consumer.py</code>	1	4
Modificar el fichero <code>extraOpt.py</code>	2	4
Modificar el fichero <code>imageProcesor.py</code>	1	4
Modificar el fichero <code>fishubuia.py</code>	2	4
Modificar el fichero y crear nuevas funciones en <code>PosicionVF.py</code>	1	1
Crear nuevos conjuntos de datos desechando <i>frame</i>	2	2
Ejecutar múltiples pruebas con conjuntos de datos más pequeños	5	8
Solucionar nuevos errores con <i>Docker</i>	4	6
Solucionar errores al intentar abrir ficheros	3	2
Redactar objetivos del proyecto en la memoria	1	1
Investigar distintas librerías para desarrollar una aplicación de escritorio	5	6

Tabla A.7: Tareas del *sprint 7*.

La mayor dificultad de este *sprint* fue la modificación de los ficheros ya existentes. Al hacer modificaciones sobre ellos había que restaurar las imágenes *Docker* haciendo que las distintas modificaciones conllevasen una gran cantidad de tiempo.

## Sprint 8: 27/05/2022 - 16/06/2022

El *sprint 8* consistió prioritariamente en la realización de la aplicación de escritorio y en la investigación sobre como clasificar un ejercicio según con que parte del cuerpo es realizado.

Tareas	Est.	Final
Desarrollar un prototipo de aplicación de escritorio	5	9
Crear un modo ejemplo en la aplicación de escritorio	4	6
Analizar la librería <i>ffmpeg</i> para el recorte de los videos	3	1
Crear videos finales para subir a la aplicación de escritorio	1	1
Realizar diferentes imágenes con Detectron2 para la memoria	2	1
Realizar diferentes imágenes para la memoria	3	4
Crear ficheros con extensión .py con las nuevas funcionalidades	2	3
Investigar como identificar movimientos superiores e inferiores	7	9
Crear un cuaderno con las posibles formas de detectar ejercicios	2	2
Redactar <i>Técnicas y herramientas</i>	4	5

Tabla A.8: Tareas del *sprint 8*.

La mayor dificultad de este *sprint* fue desarrollar un prototipo viable de aplicación de escritorio y llevarlo a la práctica.

### Sprint 9: 17/06/2022 - 07/07/2022

El *sprint 9* consistió en darle el acabado final al proyecto. Se tuvo que organizar todo el contenido, redactar la memoria, anexos, y crear una serie de vídeos para la presentación.

Tareas	Est.	Final
Mejora las funcionalidades de la aplicación de escritorio	5	9
Mejorar el diseño de la aplicación de escritorio	4	6
Redactar <i>Trabajos relacionados</i>	3	3
Redactar <i>Aspectos relevantes</i>	8	12
Redactar <i>Conclusiones y líneas futuras</i>	2	3
Redactar los anexos	5	6
Crear web de resultados	1	1
Crear diferentes videos para la presentación	1	1

Tabla A.9: Tareas del *sprint 9*.

La mayor dificultad de este *sprint* fue la realización de la memoria. En ello se invirtió mucho tiempo en la creación diversos ejemplos y en la redacción de todos los problemas surgidos, la resolución de los mismos y los resultados obtenidos.

## A.3. Estudio de viabilidad

En este apartado se va a comentar la viabilidad del proyecto. Por una parte se redactará la *viabilidad económica* que hará referencia al coste que supondría el desarrollo del proyecto, y por otra parte, se redactará la *viabilidad legal* de las librerías y herramientas utilizadas.

Estos apartados contarán con información extraída de los proyectos de [José Luis Garrido Labrador](#) y [José Miguel Ramírez Sanz](#)<sup>3</sup> ya que este proyecto es una continuación del suyo y cuenta con grandes apartados en común.

### Viabilidad económica

En este apartado se encuentran los cálculos económicos para desarrollar el proyecto. Para realizar el cálculo se han dividido los gastos en:

1. **Coste de personal:** en la tabla A.10 se encuentra una estimación de los costes que supondría contratar a un trabajador durante seis meses a jornada completa.
2. **Coste hardware:** la tabla A.11 contiene las inversiones en *hardware* y *MainFrames* necesarios para el proyecto.
3. **Coste de servicios:** la tabla A.12 contiene el conjunto de servicios necesarios para el correcto funcionamiento del proyecto.

Finalmente, en la tabla A.13 se puede observar un cálculo final del coste del proyecto.

Concepto	Coste (€)
Salario mensual bruto [3]	2.047,78
Seguridad Social (30,04 %)	615,15
Retención IRPF (30 %)	421,191
Salario mensual neto	1.403,97
<b>Total 6 meses y dos empleados</b>	<b>24.573,36</b>

Tabla A.10: Costes de personal.

---

<sup>3</sup><https://github.com/jlgarridol/TFM-FIS-IF.git>  
<https://github.com/Josemi/TFM-FIS-IA.git>

Concepto	Coste (€)	Coste amortizado (€)
Ordenador de desarrollo (x2)	950	59,37
Dispositivos paciente (x9)	100	6,25
Webcam pacientes (x9)	150	9,38
<i>MainFrame Gamma</i>	3.000	187,5
Gamma GPU (x3)	1.500	93,75
<i>MainFrame Alpha</i>	2.000	125
<b>Total 6 meses y dos empleados</b>	<b>13.650</b>	<b>853,16</b>

Tabla A.11: Costes de *hardware*.

Concepto	Coste (€)
Suscripción <i>Ngrok</i>	7,33
Lineas <i>Vodafone</i>	30
<b>Total ( por 6 meses)</b>	<b>763,98</b>

Tabla A.12: Costes de servicios.

Concepto	Coste (€)
Personal	24.573,36
<i>Hardware</i>	13.650
Servicios	763,98
<b>Total ( por 6 meses)</b>	<b>38.987,34</b>

Tabla A.13: Costes finales.

## Viabilidad legal

En este subapartado se van a exponer las distintas licencias que tienen las herramientas y librerías, así como la licencia final con la que cuenta este proyecto.

En la tabla A.14 se muestran las distintas librerías y herramientas utilizadas para la realización del proyecto y su correspondiente licencia.

Librería-Herramienta	Licencia
<i>Numpy</i>	BSD 3
<i>Pandas</i>	BSD 3
<i>OpenCV</i>	BSD 3
<i>Pynvml</i>	BSD 3
<i>Matplotlib</i>	PSF
<i>Seaborn</i>	BSD 3
<i>Plotly</i>	MIT
<i>Torchvision</i>	BSD 3
<i>Python</i>	PSF
<i>Detectron2</i>	Apache 2.0
<i>Tslearn</i>	BSD 2
<i>Dtaidistance</i>	Apache 2.0
<i>Tkinter</i>	BSD 3
<i>Scipy</i>	BSD

Tabla A.14: Tabla con las licencias de las librerías y herramientas utilizadas.

La licencia final escogida para este proyecto ha sido **GPL v3.0** ya que con esta licencia se puede utilizar el *software* desarrollado para su uso comercial, se puede modificar las implementaciones realizadas, distribuirlas, realizar patentes sobre ellas y usarlas de forma privada.

Por otra parte, hay que destacar que este proyecto se ha realizado con la ayuda de software de terceros con licencias propias que influyen sobre la viabilidad legal del proyecto.

### ***Copyright* de terceros**

Este apartado ha sido extraido del proyecto de Jose Luis Garrido Labrador.

#### **Apache 2.0:**

- *Apache Kafka - Apache Foundation*
- *Apache Zookeeper - Apache Foundation*
- *Apache Spark - Apache Foundation*
- *Docker CP - Confluentic*
- *Jitsi Meet - Jitsi*

**GPLv3:**

- *Clúster Spark Docker* - Mario Juez Gil

**BSD** todas sus variantes:

- *Caffe* - *VLV*
- *Flask* - *Pallets*
- *Jinja* - *Pallets*
- *OpenCV* - *Intel Corporation, Xperience AI*
- *Seaborn* - Michael Waskom

**MIT**

- *Bootstrap 4* - *Twitter*
- *jQuery* - *JS Foundation*

## *Apéndice B*

---

# **Especificación de Requisitos**

---

## **B.1. Introducción**

En esta sección se abordarán los diferentes objetivos y requisitos del proyecto. Se presentarán tanto los requisitos globales del proyecto, como los requisitos funcionales y casos de uso de la aplicación.

## **B.2. Objetivos generales**

- Realizar una exhaustiva investigación sobre los algoritmos de comparación de secuencias temporales, para comprobar el inicio y el final de éstas en otras secuencias.
- Investigar múltiples librerías en *Python* que permitan la localización de secuencias multidimensionales lo más parecidas posible a un patrón de referencia, dentro de una secuencia de mayor tamaño.
- Investigar y analizar como se podría clasificar un ejercicio según las partes del cuerpo que estén en movimiento.
- Recopilación de vídeos adecuados para la investigación, tanto para ser empleados en este proyecto como en proyectos futuros, comprobando que son idóneos para su análisis.
- Realizar una aplicación de escritorio por la cual se pueda observar un recorte de un ejercicio concreto de un vídeo con múltiples ejercicios.

## B.3. Catalogo de requisitos

### Requisitos del proyecto

Para ejecutar el programa correctamente se necesitan una serie de requisitos *software* y *hardware*.

#### Requisitos *software*

1. Consola Unix o macOS.
2. *Python 3*.
3. Instalación drivers de *CUDA*, versión de 9.2 a 11.2.
4. Librerías:
  - a) *Numpy*, 1.20.0.
  - b) *Pickle*, 0.7.5.
  - c) *Pandas*, 1.0.1.
  - d) *OpenCV* (cv2), 4.5.5.62.
  - e) *Garbage Collector* (gc).
  - f) Librería de manejo *NVIDIA* (pynvml), 11.4.1.
  - g) *Matplotlib*, 3.2.0.
  - h) *Seaborn*, 0.11.2.
  - i) *Plotly*.
  - j) *PyTorch* (torch), 1.9.0+cu111.
  - k) *Torchvision*, 0.10.0+cu111.
  - l) *Math*, 1.2.1.
  - m) *Os*, 3.10.5.
  - n) *tensorflow*, 2.8.0.
  - ñ) *Detectron2*, 0.6.
  - o) *pydtw*, 2.0.2.
  - p) *tslearn*, 0.5.2.
  - q) *scipy*, 1.4.1.
  - r) *Dtaidistance*, 2.3.6.

#### Requisitos *hardware*

1. Tarjeta gráfica con drivers CUDA con al menos 1GB de memoria.

## Requisitos de la aplicación

En este apartado se mostrarán los requisitos funcionales de la aplicación de escritorio. Aunque ha sido creada para comprobar los resultados de las ejecuciones, esta aplicación también podría ser usada por un terapeuta que quiere observar como de bien están realizando sus pacientes los ejercicios prescritos.

### Actores

1. *Terapeuta*: este actor será el que utilice la aplicación. Idealmente será el terapeuta correspondiente al paciente del que se están analizando los vídeos.
2. *Científico de datos*: este actor será quien manipule los ficheros que procesan las secuencias y le pasará al terapeuta los *frames* de inicio y final de la secuencia. Este actor no intervendrá con la aplicación por lo que no recibirá un diagrama de uso.
3. *Usuario*: este actor será un miembro del tribunal o cualquier otro individuo que quiera comprobar de una forma visual la finalidad del proyecto sin tener que ejecutarlo.

En este apartado se mostrarán los requisitos funcionales del proyecto la aplicación.

- **RF.1** Manipulación de vídeos.

- **RF.1.1** El terapeuta podrá seleccionar el vídeo que quiere localizar.
- **RF.1.2** El terapeuta podrá seleccionar el vídeo en el que quiere que se localice el ejercicio concreto.
- **RF.1.3** El usuario podrá seleccionar el vídeo que quiere localizar.
- **RF.1.4** El usuario podrá seleccionar el vídeo en el que quiere que se localice el ejercicio concreto.

- **RF.2** Visualización de vídeos tanto del terapeuta como del paciente.

- **RF.2.1** El terapeuta podrá reproducir el vídeo concreto.
- **RF.2.2** El terapeuta podrá reproducir el vídeo que contiene todos los ejercicios.
- **RF.2.3** El terapeuta podrá pausar el vídeo concreto.

- **RF.2.4** El terapeuta podrá pausar el vídeo que contiene todos los ejercicios.
  - **RF.2.5** El usuario podrá reproducir el vídeo concreto.
  - **RF.2.6** El usuario podrá reproducir el vídeo que contiene todos los ejercicios.
  - **RF.2.7** El usuario podrá pausar el vídeo concreto.
  - **RF.2.8** El usuario podrá pausar el vídeo que contiene todos los ejercicios.
- **RF.3** Visualización de resultados.
    - **RF.3.1** El terapeuta podrá reproducir el vídeo resultante.
    - **RF.3.2** El terapeuta podrá pausar el vídeo resultante.
    - **RF.3.3** El terapeuta podrá indicar el *frame* en el que se inicia el vídeo.
    - **RF.3.4** El terapeuta podrá indicar el *frame* en el que se finaliza el vídeo.
    - **RF.3.5** El usuario podrá reproducir el vídeo resultante.
    - **RF.3.6** El usuario podrá pausar el vídeo resultante.
  - **RF.4** Gestión de vídeos: carga, eliminación
    - **RF.4.1** El terapeuta podrá introducir en el directorio que contiene los vídeos concretos todos los vídeos que desee analizar.
    - **RF.4.2** El terapeuta podrá introducir en el directorio que contiene los vídeos completos todos los vídeos que desee analizar.
    - **RF.4.3** El usuario podrá introducir en el directorio que contiene los vídeos concretos todos los vídeos que desee analizar.
    - **RF.4.4** El usuario podrá introducir en el directorio que contiene los vídeos completos todos los vídeos que desee analizar.
  - **RF.5** Gestión de resultados: almacenamiento.
    - **RF.5.1** El terapeuta podrá guardar el vídeo resultante.
  - **RF.6** Comprobación de resultados en distintos modos de ejecución.
    - **RF.6.1** El terapeuta podrá cambiar al entre el modo de ejemplo y el modo de ejecución.

- **RF.6.2** El usuario podrá cambiar al entre el modo de ejemplo y el modo de ejecución.
- **RF.6.3** El usuario podrá observar un recorte del ejercicio sin seleccionar el *frame* de inicio.
- **RF.6.4** El usuario podrá observar un recorte del ejercicio sin seleccionar el *frame* de final.

## B.4. Especificación de requisitos

En este apartado se van a mostrar, por una parte los diagramas de casos de uso, como se puede observar en las figuras B.1 y B.2 y por otra, las tablas de casos de uso.

La única diferencia que muestra la aplicación entre el *modo ejemplo ON* y el *modo ejemplo OFF*, es que en este último caso se deberán de especificar los *frames* de inicio y de final.

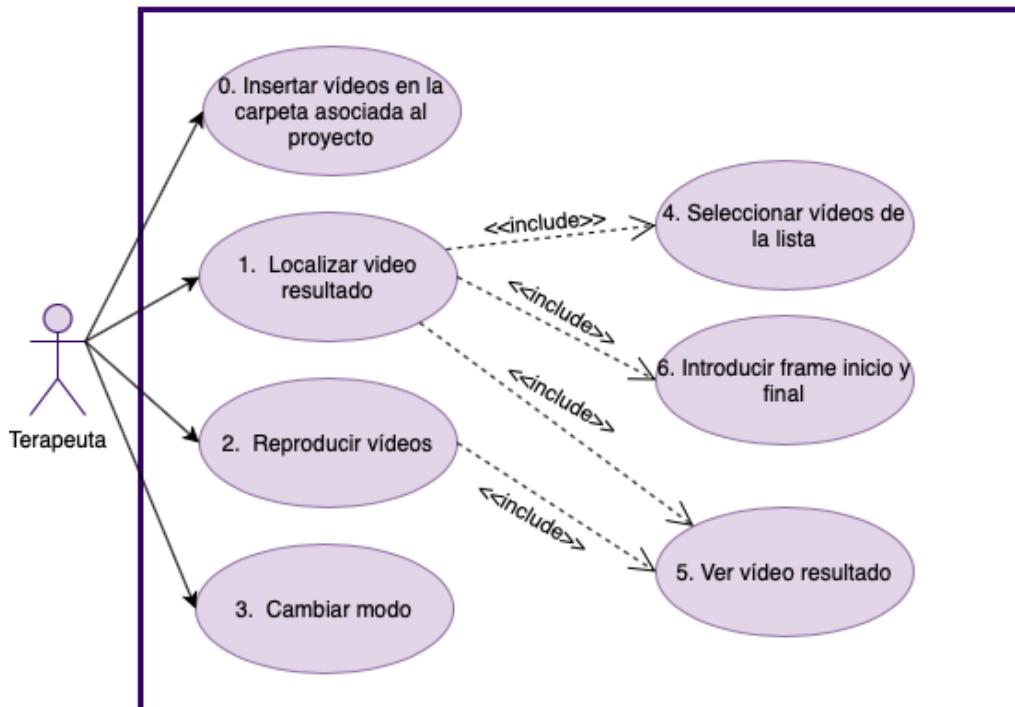


Figura B.1: Diagrama de casos de uso para el terapeuta.

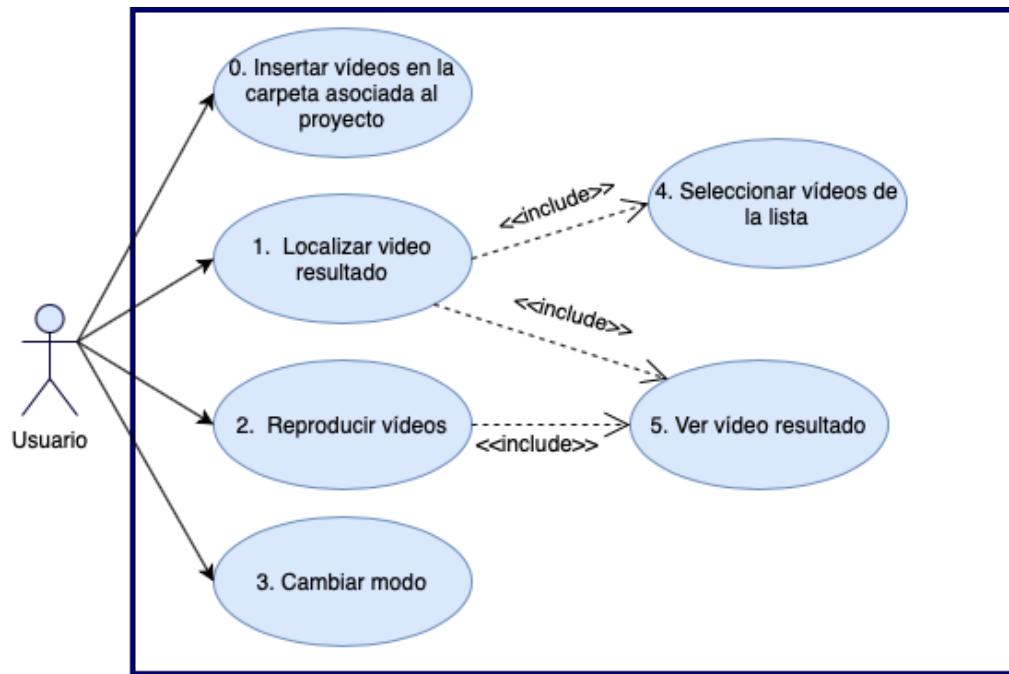


Figura B.2: Diagrama de casos de uso para el usuario.

---

Caso de uso 0: Insertar vídeos.		
Descripción	Permite al usuario cargar vídeos de su equipo a la aplicación.	
Requisitos	RF-4.1 RF-4.2 RF-4.3 RF-4.4	
Precondiciones	Ninguna	
Secuencia normal	Paso	Acción
	1	El usuario o terapeuta introducen los vídeos en el directorio deseado.
	2	Al ejecutar la aplicación los vídeos se cargan automáticamente en la barra de selección.
Postcondiciones	Vídeos aparecen en cada una de sus listas de selección para posteriormente poder ser mostrados.	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	

---

Tabla B.1: Caso de uso 0: Insertar vídeos.

---

Caso de uso 1: Localiza el vídeo.

---

Descripción	Permite al usuario localizar el vídeo concreto dentro del vídeo que contiene múltiples ejercicios.	
Requisitos	RF-1.1 RF-1.2 RF-1.3 RF-1.4	
Precondiciones	Tener por lo menos algún vídeo concreto y algún vídeo completo cargado	
Secuencia normal	Paso	Acción
	1	El usuario o terapeuta seleccionan el vídeo concreto deseado.
	2	El usuario o terapeuta seleccionan el botón mostrar.
	3	El usuario o terapeuta seleccionan el vídeo concreto deseado.
	4	El usuario o terapeuta seleccionan el botón mostrar.
	5	El usuario o terapeuta seleccionan el botón recortar.
	4	Aparece el vídeo recortado.
Postcondiciones	El vídeo localizado se reproduce automáticamente en la aplicación y se guarda una copia del mismo en la carpeta en la que se ejecuta el proyecto.	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	

---

Tabla B.2: Caso de uso 1: Localiza el vídeo.

---

Caso de uso 2: Reproducir vídeos.															
Descripción	Permite al usuario reproducir y pausar los diferentes videos.														
Requisitos	RF-2.1 RF-2.2 RF-2.3 RF-2.4 RF-2.5 RF-2.6 RF-2.7 RF-2.8														
Precondiciones	Tener el video que se desea reproducir o pausar cargado.														
Secuencia normal	<table border="1"> <thead> <tr> <th>Paso</th><th>Acción</th></tr> </thead> <tbody> <tr> <td>1</td><td>El usuario o terapeuta seleccionan un video.</td></tr> <tr> <td>2</td><td>El usuario o terapeuta pincha sobre el botón play.</td></tr> <tr> <td>3</td><td>Vídeo comienza a reproducirse.</td></tr> <tr> <td>4</td><td>El usuario o terapeuta puede pinchar sobre el botón pause.</td></tr> <tr> <td>5</td><td>El usuario o terapeuta puede pinchar sobre el botón stop.</td></tr> <tr> <td>6</td><td>El video se para.</td></tr> </tbody> </table>	Paso	Acción	1	El usuario o terapeuta seleccionan un video.	2	El usuario o terapeuta pincha sobre el botón play.	3	Vídeo comienza a reproducirse.	4	El usuario o terapeuta puede pinchar sobre el botón pause.	5	El usuario o terapeuta puede pinchar sobre el botón stop.	6	El video se para.
Paso	Acción														
1	El usuario o terapeuta seleccionan un video.														
2	El usuario o terapeuta pincha sobre el botón play.														
3	Vídeo comienza a reproducirse.														
4	El usuario o terapeuta puede pinchar sobre el botón pause.														
5	El usuario o terapeuta puede pinchar sobre el botón stop.														
6	El video se para.														
Postcondiciones	Ninguna.														
Excepciones	Ninguna														
Importancia	Alta														
Urgencia	Alta														

---

Tabla B.3: Caso de uso 2: Reproducir vídeos.

---

Caso de uso 3: Cambiar modo.

---

Descripción	Permite al usuario o al terapeuta cambiar el modo de la aplicación.	
Requisitos	RF-6.1 RF-6.2 RF-6.3 RF-6.4	
Precondiciones	Ninguna	
Secuencia normal	Paso	Acción
	1	El usuario o terapeuta seleccionan el botón de cambio de modo.
	2	Se cambia el modo de la pantalla.
Postcondiciones	Se habilitan o deshabilitan las casillas de los <i>frames</i> , se actualizan los vídeos y se cambian los colores.	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	

---

Tabla B.4: Caso de uso 3: Cambiar modo.

Caso de uso 4: Seleccionar vídeos de la lista.		
Descripción	Permite al usuario o terapeuta seleccionar los vídeos concretos o completos que desea reproducir.	
Requisitos	RF-1.2 RF-1.2 RF-1.3 RF-1.4	
Precondiciones	Que la lista contenga vídeos.	
Secuencia normal	Paso	Acción
	1	El usuario o terapeuta seleccionarán un vídeo concreto de la lista.
	2	El usuario o terapeuta pincha sobre el botón añadir.
	3	El vídeo se carga y se reproduce automáticamente.
	4	El usuario o terapeuta seleccionarán un vídeo completo de la lista.
	5	El usuario o terapeuta pincha sobre el botón añadir.
	6	El vídeo se carga y se reproduce automáticamente.
Postcondiciones	Se muestran ambos vídeos cargados y reproduciéndose automáticamente hasta su finalización.	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	

Tabla B.5: Caso de uso 4: Seleccionar vídeos de la lista.

---

Caso de uso 5: Ver vídeo resultado.

---

Descripción	Permite al usuario o terapeuta ver el vídeo final generado.	
Requisitos	RF-6.3	
	RF-6.4	
	RF-5.1	
	RF-3.1	
	RF-3.2	
	RF-3.5	
Precondiciones	RF-3.6	
	Haber seleccionado tanto un vídeo concreto como completo. En el caso de no estar en el modo ejemplo será necesario indicar los <i>frames</i> por los que se recorta.	
Secuencia normal	Paso	Acción
	1	El usuario o terapeuta seleccionarán un vídeo concreto de la lista.
	2	El usuario o terapeuta seleccionarán un vídeo completo de la lista.
	3	En caso de que sea necesario se especifican los <i>frames</i> de inicio y final.
	4	Se pulsa el botón recortar.
	5	El vídeo recortado se guarda automáticamente.
Postcondiciones	El vídeo recortado se muestra en la aplicación y se reproduce automáticamente.	
Excepciones	En caso de que sea necesario especificar el <i>frame</i> de inicio y final y no se haya especificado informará del error.	
Importancia	Alta	
Urgencia	Alta	

---

Tabla B.6: Caso de uso 5: Ver vídeo resultado.

---

Caso de uso 6: Introducir <i>frames</i> de inicio y final.		
Descripción	Permite al terapeuta indicar el <i>frame</i> de inicio y final del vídeo.	
Requisitos	RF-3.3 RF-3.4	
Precondiciones	Haber seleccionado tanto un vídeo concreto como completo.	
Secuencia normal	Paso	Acción
	1	El terapeuta seleccionarán un vídeo concreto de la lista.
	2	El terapeuta seleccionarán un vídeo completo de la lista.
	3	El terapeuta especifica los <i>frames</i> de inicio y final.
	4	Se pulsa el botón recortar.
	5	El vídeo recortado se guarda automáticamente.
Postcondiciones	El vídeo recortado se muestra en la aplicación y se reproduce automáticamente.	
Excepciones	Ninguna.	
Importancia	Alta	
Urgencia	Alta	

---

Tabla B.7: Caso de uso 6: Introducir *frames* de inicio y final.



## *Apéndice C*

---

# Especificación de diseño

---

## C.1. Introducción

La fase de diseño permite planificar un proyecto para su correcta implementación, desarrollo y evolución. En este apartado se van a exponer los diferentes diseños que se han llevado a cabo para obtener unas buenas soluciones a los problemas planteados. Una de las partes más importantes de este proyecto ha sido la investigación, por lo que el diseño en la mayor parte de este proyecto no ha sido necesaria. Los diseños realizados han sido:

1. **Diseño de datos:** en esta parte del diseño se mostrarán todas las estructuras de datos utilizadas y una serie de diagramas para entender la estructura del proyecto.
2. **Diseño procedimental:** en esta parte del diseño se mostrará principalmente la comunicación entre el flujo y las implementaciones realizadas.
3. **Diseño arquitectónico:** en esta parte del diseño se especificará la arquitectura del proyecto.

En todos los apartados anteriormente mencionados se expondrá, tanto las fases de diseño de la aplicación, como las fases de diseño del proyecto.

## C.2. Diseño de datos

Este apartado cuenta con alguna parte del *Diseño de datos* de José Miguel Ramírez Sanz

### Diseño de datos del proyecto

Este proyecto es mayoritariamente un proyecto de investigación por lo que apenas tiene diseño de datos. En este apartado se va a mostrar un diagrama de clases, como se puede observar en la figura C.1, con la implementación final de la clase `PosicionVF` almacenada en el directorio `src/process`. El diagrama se ha obtenido del proyecto de José Miguel Ramírez Sanz y se le han añadido las nuevas funcionalidades creadas que son las siguientes:

1. `devuelveAngulos()`: esta función devuelve una matriz con todos los ángulos que componen el esqueleto analizado.
2. `devuelvePos()`: esta función devuelve una matriz con todos las posiciones relativas al esqueleto y los ángulos calculados. A estos últimos valores se les añadirá una dimensión extra para subsanar los fallos que genera comparar secuencias con distintas dimensiones.
3. `devuelvePosSuperiores()`: esta función devuelve una matriz que contiene las posiciones de la nariz, hombro izquierdo, hombro derecho, cuello, ángulo del cuello izquierdo, ángulo del cuello derecho, codo izquierdo, codo derecho, mano izquierda, mano derecha, ángulo del codo izquierdo, ángulo del codo derecho, ángulo del hombro izquierdo y ángulo del hombro derecho.
4. `devuelvePosInferiores()`: esta función devuelve una matriz que contiene las posiciones de la cadera izquierda, cadera derecha, rodilla izquierda, rodilla derecha, ángulo izquierdo de la cadera, ángulo derecho de la cadera, tobillo izquierdo, tobillo derecho, ángulo izquierdo de la rodilla y ángulo derecho de la rodilla.

Por otro lado, la clase `Interfaz` es la que se comunica con el flujo de datos (de ahí su nombre), en ella se carga el modelo para posteriormente crean las posiciones.

Además, en el diagrama se pueden observar las relaciones entre las clases más usadas en el código. En este diagrama se puede observar también como se incorpora la herramienta *Detectron2* en el proyecto (clases `cfg`, `model_zoo`, `Instances` y `DefaultPredictor`).

Por otra parte, los ficheros generados para localizar y clasificar la secuencia, localizados en `src/scripts/deploy` son demasiado sencillos como para generar con ellos un diagrama de clases. Esos ficheros cuentan con las siguientes características:

1. `find_frames.py`: este fichero hay que ejecutarle pasándole como parámetros dos secuencias. La primera de ellas corresponderá a la secuencia de ejercicios concretos y la segunda a la secuencia compuesta por múltiples ejercicios. Una vez recibe las dos secuencias, extrae las posiciones de los archivos con extensión `.pickle` y ejecuta la búsqueda de secuencias. Finalmente informa al usuario de los *frames* de inicio y final de la secuencia. Adicionalmente se han implementado las funciones que clasifican el ejercicio para informar de que tipo es el ejercicio concreto.
2. `classify_exercises.py`: este fichero hay que ejecutarle pasándole como parámetros la secuencia del ejercicio concreto y devolverá una clasificación en función de si se trata de un ejercicio sobre las partes superiores o inferiores del cuerpo humano.

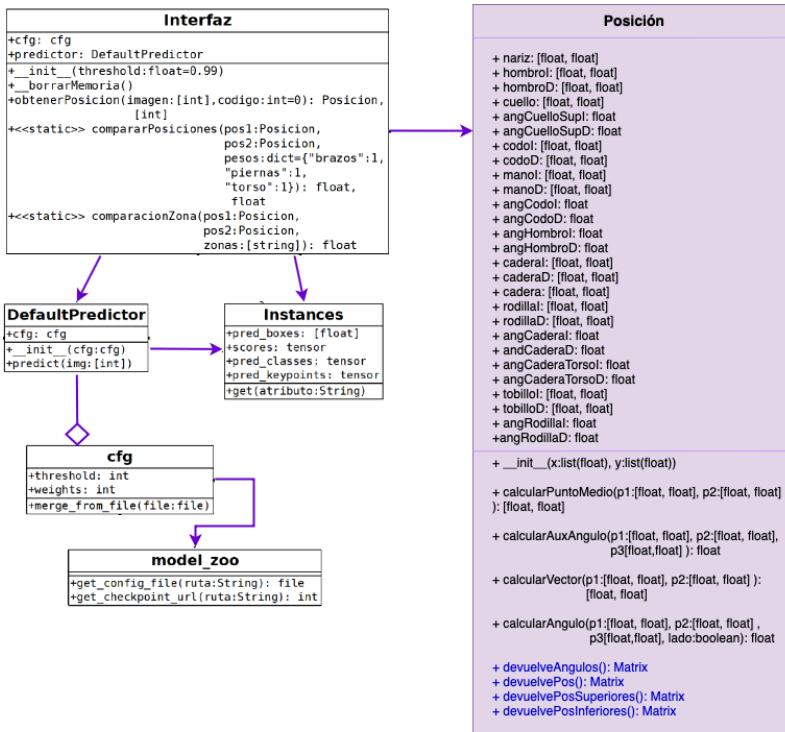


Figura C.1: Diagrama de clases del proyecto.

## Diseño de datos de la aplicación

En este apartado se va a mostrar un diagrama de clases para la aplicación de escritorio, como se puede observar en la figura C.2, en este diagrama se podrán observar las siguientes clases almacenadas en `app/`:

1. **Window** dentro de `main.py`, esta clase contiene toda la estructura gráfica de la ventana de la aplicación. En esta clase se pueden apreciar todas las inicializaciones de *frames*, *labels*, *botones* ... y las variables que cada uno precisa. Todos estos valores serán almacenados como atributos para poder acceder a ellos con posterioridad desde el resto de clases.
2. **MiddleWindow** dentro de `MiddleWindow.py`, esta clase tiene como atributo la ventana de inicio. Es la clase encargada de implementar las funciones básicas que debe desarrollar la aplicación, como por ejemplo, mostrar, reproducir o pausar un vídeo, cambiar los colores y habilitar o deshabilitar funcionalidades al cambiar de modo, etc.
3. **PredictWindow** dentro de `PredictWindow.py`, al igual que la clase anterior, el único atributo que posee es la ventana de inicio. Es la clase encargada de generar el recorte del vídeo, y tiene en cuenta el modo en el que se está trabajando.

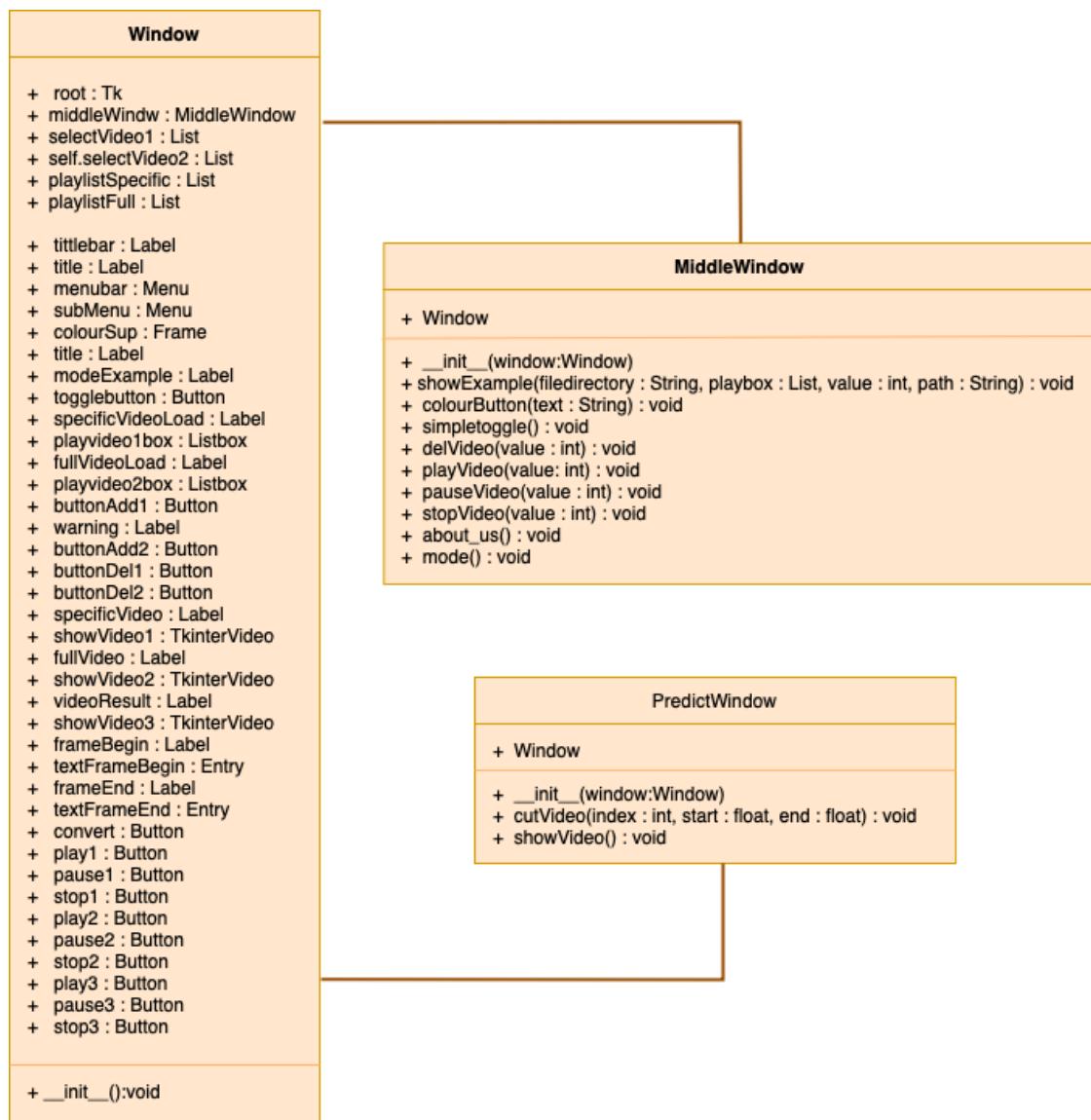


Figura C.2: Diagrama de clases de la aplicación.

### C.3. Diseño procedimental

Este apartado es una ampliación del Diseño de datos de José Luis Garrido Labrador

#### Diseño procedural del proyecto

En este apartado se van a mostrar un par de diagramas de secuencias. En la figura C.3, se muestra el proceso general de la aplicación. Este programa contiene una clase **Ingestor** que se encarga de encolar los *frames* recibidos, y procesarlos. En este proyecto los *frames* vienen procedentes de un único vídeo y lo que programa realiza es ejecutar ese vídeo en bucle por lo que el proceso únicamente acabaría cuando el usuario finaliza el programa.

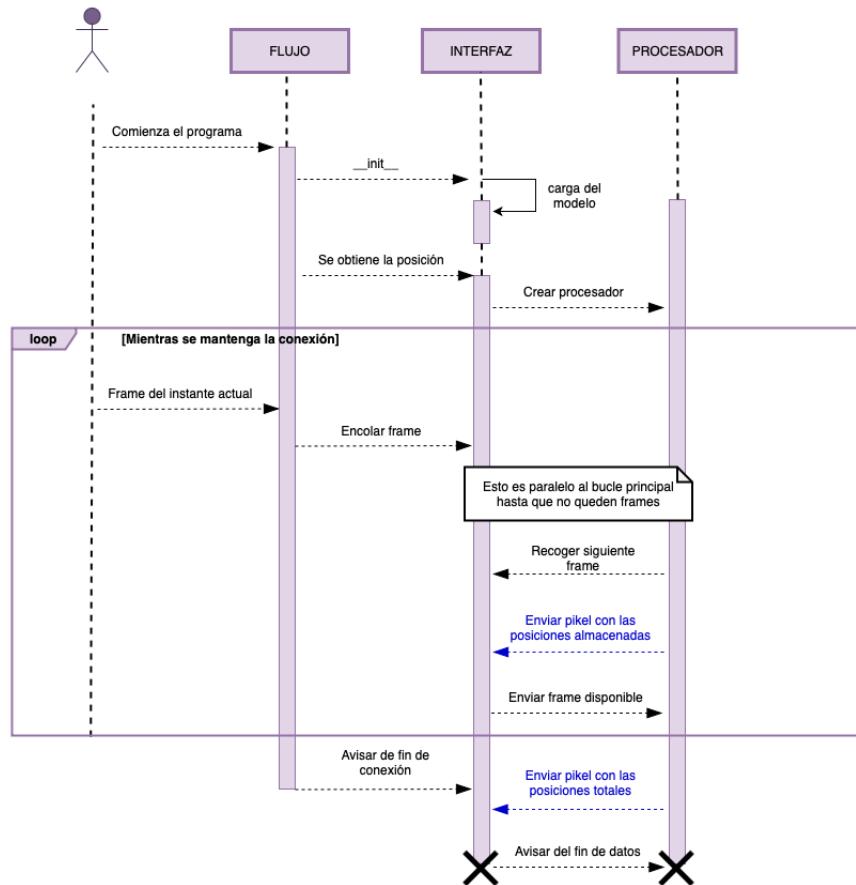


Figura C.3: Diagrama de secuencias de la parte encargada de obtener las posiciones.

Por otra parte, en la figura C.4 se puede observar el proceso que se genera para obtener los *frames* de inicio y final y la clasificación del ejercicio.

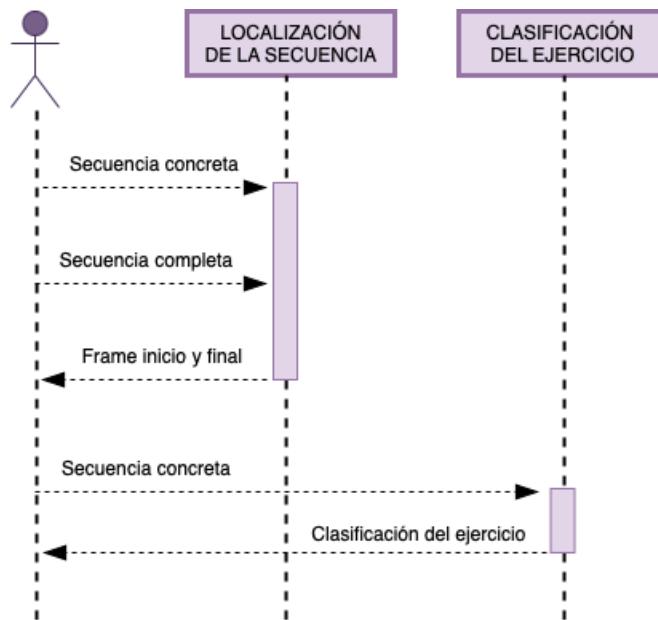


Figura C.4: Diagrama de secuencias de la parte encargada de localizar las secuencias.

## Diseño procedimental de la aplicación

En el diagrama de secuencias que se puede apreciar en la figura C.5, se muestran las tareas principales de la aplicación de escritorio: cargar, reproducir, pausar y detener un vídeo, cambiar el modo de la aplicación y generar un recorte del vídeo que contiene la secuencia de ejercicios completa.

Como se puede apreciar la única diferencia entre el *modo ejemplo ON* y el *modo ejemplo OFF* es que en el segundo hay que especificar los *frames* por los que se desea recortar el vídeo, mientras que en el caso anterior lo hace automáticamente.

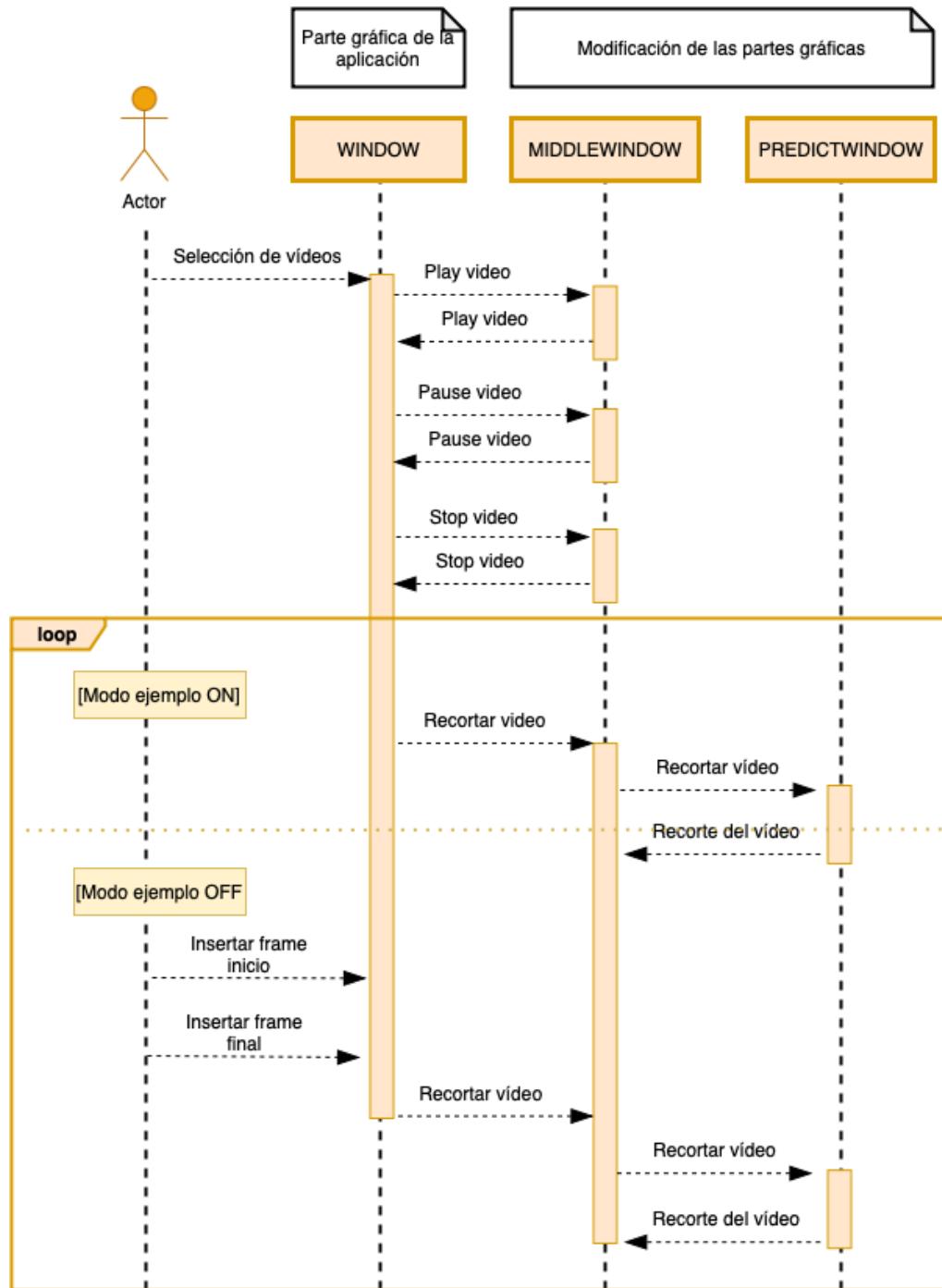


Figura C.5: Diagrama de secuencias de la aplicación.

## C.4. Diseño arquitectónico

Este apartado ha sido extraído del *Diseño de datos arquitectónicos* de José Luis Garrido Labrador

### Diseño arquitectónico del proyecto

Una parte muy esencial de este proyecto es el uso de y despliegue de máquinas virtuales *Docker*. Concretamente hay cuatro tipos de imágenes *Docker*. La primera se encarga de la serialización de los *frames* y lanzan el script de *Python* de encolado, esta máquina se crea y destruye a voluntad de las conexiones de los pacientes. La segunda y tercera imágenes son el servicio de *Zookeeper* y de *Kafka*. Estas imágenes no se duplican en caso de cambios en las conexiones, únicamente se crean o borran colas. Por último, la cuarta imagen es la transformación del flujo. Se crean o se destruyen según las conexiones de los pacientes y se parametrizan para que consuman un flujo concreto y hagan un procesamiento concreto. El diagrama de despliegue de las máquinas virtuales *docker* se puede observar en la C.6.

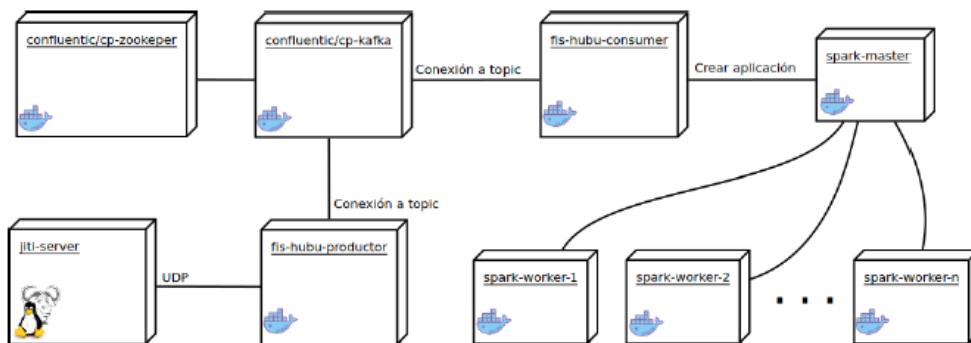


Figura C.6: Diagrama de despliegue de las máquinas virtuales *docker*.

Por otra parte, se ha creado un diagrama con la estructura de los directorios más relevantes del proyecto y se puede observar en la figura C.7. En esta estructura destacan los directorios:

1. **src/scripts/deploy**: contiene todos los ficheros necesarios para desplegar los servidores e imágenes *Docker*. Además cuenta con los ficheros específicos de este proyecto que localizan y clasifican las secuencias.
2. **src/process**: contiene todos los ficheros necesarios para la obtención de las secuencias.

3. **src/pruebas:** contiene los distintos *notebooks* generados para explicar las fases del proyecto, a si como muchas de las secuencias generadas y vídeos realizados.

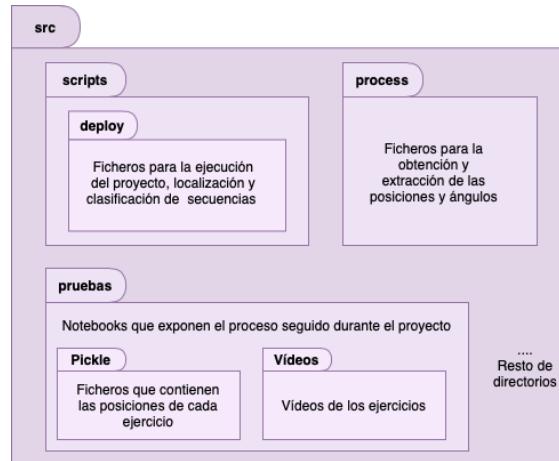


Figura C.7: Diagrama de paquetes de la aplicación.

## Diseño arquitectónico de la aplicación

En este apartado se definirá la estructura en paquetes de la aplicación. Como se puede observar en la figura C.8 la distribución es muy simple. Se almacenan los vídeos en diferentes directorios dentro de la carpeta en la que se encuentran las clases de dicha aplicación. Según el directorio en el que se encuentren cada uno de los vídeos, se cargarán en una sección u otra de la aplicación.

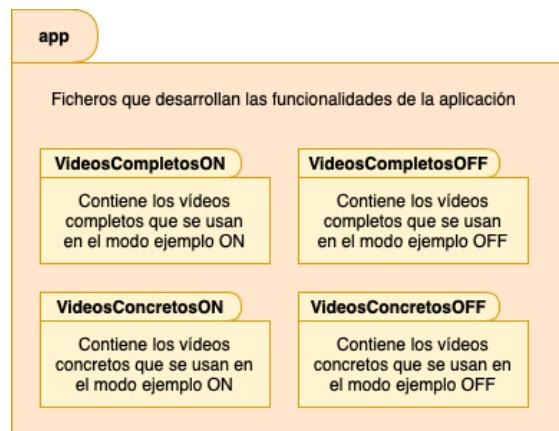


Figura C.8: Diagrama de paquetes de la aplicación.

## *Apéndice D*

---

# **Documentación técnica de programación**

---

## **D.1. Introducción**

En este apartado se mostrarán los distintos requisitos e instrucciones necesarias para la instalación y comprensión del proyecto. Para ello se deberá tener en cuenta:

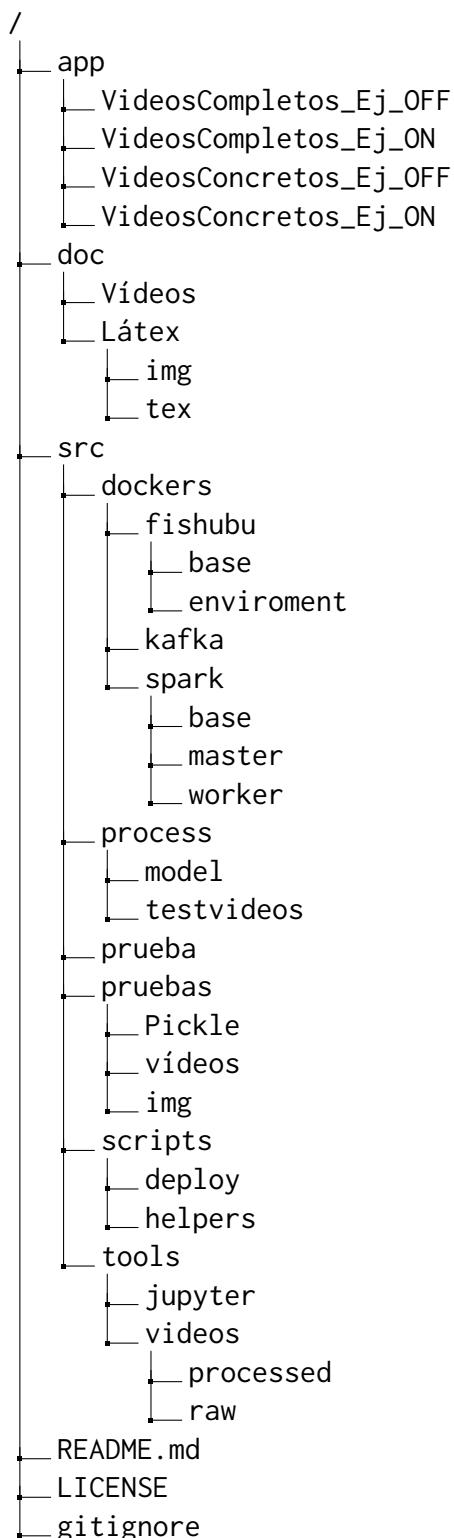
1. **Estructura de directorios:** se mostrará la estructura general del proyecto. Este apartado también se puede observar en el repositorio de *GitHub* que contendrá, a parte de la estructura de directorios, las tareas realizadas.
2. **Manual del programador:** manual en el que se exponen los pasos a seguir para poder continuar con el proyecto.
3. **Compilación, instalación y ejecución del proyecto:** manual que abarca las instrucciones básicas para poder ejecutar el proyecto.
4. **Pruebas del sistema,** explicación de las distintas pruebas realizadas y de como lanzarlas.

## **D.2. Estructura de directorios**

En este subapartado se comentará la estructura del contenido del proyecto. Es la misma que la que se puede apreciar en el repositorio de *GitHub*<sup>1</sup>:

---

<sup>1</sup><https://github.com/lnc1002/TFG-Evaluacion-Ejercicios-Rehabilitacion.git>

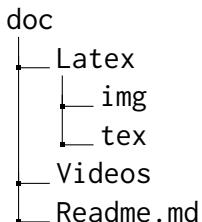


Como se puede observar, de la raíz cuelgan las carpetas `app`, `src`, `doc` y tres documentos con las siguientes funcionalidades:

1. `README.md`: documento que se encuentra en la raíz del repositorio de *GitHub* y aparecerá en más directorios. Este fichero muestra información relativa al proyecto y a los ficheros que contiene el directorio en el que se ubica. Está compuesto por un resumen y unas instrucciones básicas de como está estructurado el contenido del proyecto.
2. `LICENSE`: documento que contiene la licencia del proyecto en *GitHub*.
3. `.gitignore`: documento que especifica que archivos o directorios no se desean subir al repositorio.

## Documentación

Una vez comentada la estructura de ficheros se va a exponer la estructura de los directorios que cuelgan desde la raíz del proyecto. La carpeta `doc` contiene toda la documentación del proyecto y tiene la siguiente estructura:



Dentro del directorio `Latex` se encuentra toda la documentación de memoria y anexos realizada sobre L<sup>A</sup>T<sub>E</sub>X. En la carpeta `/doc/Latex/img` se localizan todas las imágenes y diagramas usados en este proyecto. Tanto una gran cantidad de imágenes como de diagramas han sido realizados con la herramienta *Diagrams.net* [1].

En el directorio `Videos` se encuentran unos breves vídeos del funcionamiento de como ejecutar el proyecto, obtener los *frames* resultantes a partir de los *pickle* generados y como interactuar con la aplicación de escritorio.

## Código

La carpeta `src` contiene todos los ficheros y directorios necesarios tanto para ejecutar el programa como para la localización de las secuencias. En concreto vamos a exponer el contenido del directorio `/src/scripts/deploy/`:

```

src
└── scripts
    └── deploy
        ├── new-stream
        ├── start-server
        ├── stop-server
        ├── stop-stream
        ├── classify_exercises.py
        └── find_frames.py

```

Los dos últimos ficheros son los que se han implementado para poder localizar la secuencia o clasificar el ejercicio una vez haya terminado el proceso de extracción de posiciones.

Por otra parte, el directorio `/src/pruebas/` contiene:

```

src
└── pruebas
    ├── C1_Pruetas_paquetes_DTW.ipynb
    ├── C2_Buscar_Subsecuencia.ipynb
    ├── C3_Busqueda_y_agrupacion_de_secuencia_SIMILAR.ipynb
    ├── C4_Busqueda_multiples_secuencias.ipynb
    ├── C5_Busqueda_inicio_y_final.ipynb
    ├── C6_Pruetas_busqueda_angulos.ipynb
    ├── C7_Pruetas_busqueda_posiciones.ipynb
    ├── C8_Pruetas_recortando_frames.ipynb
    ├── C9_Pruetas_clasificacion_movimientos.ipynb
    └── C10_Resultados_finales.ipynb
    ├── Pickle
    │   ├── Posiciones_inferiores
    │   ├── Posiciones_superiores
    │   ├── Posiciones_completas
    │   └── Angulos_completos
    ├── vídeos
    ├── img
    └── README.md

```

Como se puede observar dentro de este directorio se encuentran los diferentes *notebooks* que se han empleado para mostrar las pruebas realizadas durante todo el proceso del proyecto. Además en el directorio `/src/pruebas/Pickle` se encuentran almacenadas gran parte de las secuencias generadas y en el directorio `/src/pruebas/vídeos` algunos de los vídeos creados para dichas pruebas.

## Aplicación

La carpeta **app** contiene todos los ficheros y directorios necesarios para poder ejecutar la aplicación de escritorio y su estructura es la siguiente:

```
app
├── VideosCompletos_Ej_OFF
├── VideosCompletos_Ej_ON
├── VideosConcretos_Ej_OFF
├── VideosConcretos_Ej_ON
├── .gitignore
├── MiddleWindow.py
├── PredictWindow.py
├── main.py
└── README.md
└── requirements.txt
```

## D.3. Manual del programador

Todo el proyecto ha corrido sobre la máquina *gamma* del equipo *ADMIRABLE* de la Universidad de Burgos y para poder llevar a cabo las ejecuciones se han tenido que instalar una serie de programas sobre el entorno de *Ubuntu*. Muchos de estos programas han sido instalados mediante el comando *wget* y el entorno final ha quedado con los siguientes programas y versiones:

1. **Anaconda**: es una distribución de *Python* utilizada para el desarrollo científico tanto gráfico como analítico [6]. En este proyecto se ha utilizado la versión **4.11.0**.
2. **NVIDIA**: es un *software* que ofrece una amplia gama de posibilidades para la construcción o aceleración de aplicaciones [4]. En este proyecto se ha utilizado la versión **460.80**.

3. **CUDA**: es una plataforma de computación paralela y un modelo de programación desarrollados por *NVIDIA* [5]. En este proyecto se ha utilizado la versión de CUDA **11.2**.
4. **Detectron2**: es una plataforma para la detección de objetos [9]. En este proyecto se ha utilizado la versión 0.6. Para más información sobre *Detectron2* visitar el apartado de la memoria *Técnicas y herramientas*.
5. **Tensorflow**: es una plataforma de código abierto que cuenta con una amplia cantidad de herramientas y recursos para el aprendizaje automático [7]. En este proyecto se ha utilizado la versión **2.8.0**.

## *Script* de despliegue

Este apartado ha sido extraído del *Script* de despliegue de José Luis Garrido Labrador

Para el despliegue de los servicios mediante *Docker* se han creado cuatro *scripts* localizados en la carpeta `src/scripts/deploy`. Estos códigos en *Bash* son los siguientes:

1. **start-server**: se encarga de instanciar los diferentes servicios.

```

1 Sintaxis
2     start-server <Carpeta de salida> <N. de CPU
                  master> <N. de workers> <N. de CPU por
                  worker> <Memoria por worker>

```

2. **stop-server**: detiene todos los servicios.

```

1 Sintaxis
2     stop-server

```

3. **new-stream**: genera un nuevo flujo completo que se va a procesar. El funcionamiento es transparente. Recibe los parámetros para el `emitter.py` y para el `consumer.py`. Por seguridad es preferible solamente modificar los parámetros de gestión del flujo y no los de comunicación. Devuelve el identificador del flujo. Es importante este valor para poder cerrarlo después.

```
1 Sintaxis
2     new-stream "Parámetros_emitter.py" "Parámetros_
    consumer.py"
3     # Es muy importante mantener las comillas
```

4. **stop-stream**: detiene todos los procesos sobre un flujo concreto.

```
1 Sintaxis
2     stop-stream <ID del flujo>
```

## Script de análisis

Para la comparación de secuencias se han creado dos *scripts*:

1. **find\_frames.py**: localiza el patrón de referencia dentro de la secuencia de mayor tamaño y devolverá los *frames* en los que se ha iniciado y finalizado la supuesta secuencia encontrada. Por último devolverá también una clasificación del tipo de ejercicio realizado.

```
1 Sintaxis
2     find_frames.py <secuencia del ejercicio
    concreto> <secuencia con varios ejercicios>
```

2. **classify\_exercises.py**: informará del tipo de ejercicio que se está realizando en el vídeo pasado como argumento.

```
1 Sintaxis
2     classify_exercises.py <secuencia del ejercicio
    concreto>
```

## D.4. Compilación, instalación y ejecución del proyecto

Parte de este apartado ha sido extraído del apartado de *Compilación, instalación y ejecución del proyecto* de José Luis Garrido Labrador

Como se ha mencionado anteriormente, en la carpeta `src/scripts/deploy` se encuentran los *scripts* para instalar el proyecto y poder ejecutarlo.

Para que los *scripts* se ejecuten correctamente es necesario que se corran sobre un sistema operativo GNU/Linux con el servicio *Docker* y la extensión *Nvidia container toolkit* [2] instalados. Adicionalmente se necesitará que la máquina donde se vayan a ejecutar los *workers* disponga de una tarjeta gráfica *Nvidia* instalada con soporte para *CUDA 10.2*.

Es posible que sea necesario cambiar el fichero *Dockerfile* de la carpeta `dockers/fishubu/base` para que use los drivers de la tarjeta gráfica del equipo sobre el que se lanza la imagen *docker*.

Antes de lanzar los servicios es importante haber creado previamente las imágenes maestras (**orden build**) para los diferentes *dockers*. Desde la carpeta `deploy`:

```

1 docker build -f ../../dockers/fishubu/base/Dockerfile -t
   fishubu-base:1.0.0 ../../
2 docker build -f
   ../../dockers/fishubu/enviroment/Dockerfile -t
   fishubu-env:1.0.0 ../../
3 docker build ../../dockers/spark/base -t
   spark-base-fis:2.4.5
4 docker build ../../dockers/spark/master -t
   spark-master-fis:2.4.5
5 docker build ../../dockers/spark/worker -t
   spark-worker-fis:2.4.5

```

El orden de ejecución de los *scripts* es el siguiente:

1. Ejecutar **start-server** para que los servicios de *Kafka* y *Spark* estén activos y den soporte a los flujos que lo necesiten.

2. **Ejecutar `new-stream`** recibiendo como parámetros la configuración deseada para el emisor y para el consumidor. Este devuelve el identificador del flujo, será necesario para pedir el cierre del flujo.

Para detener el flujo los pasos son los siguientes:

1. **Ejecutar `stop-stream`** para cada flujo arrancado.
2. **Ejecutar `stop-server`** y se detendrán todos los servicios.

Para localizar la secuencia y clasificar los ejercicios el orden en el que se realicen las ejecuciones es indiferente. Por otra parte, tampoco es necesario que se esté corriendo el programa que extrae las secuencias, simplemente se necesitará tener algunas de ellas almacenadas:

1. **Ejecutar `find_frames.py`** para la extracción de la secuencia.
2. **Ejecutar `classify_exercises.py`** para la clasificación del ejercicio.

## D.5. Fallos y soluciones

En este apartado se van a exponer algunos de los errores por los que se ha visto comprometido el proyecto, sus causas y sus posibles soluciones.

Los fallos que producidos durante el trascurso de las ejecuciones son almacenados en diferentes ficheros. Por una parte se encuentran los ficheros de ***log*** ubicados en el directorio `/tmp/fishubulogs`. Para consultar estos ficheros será necesario indicar el identificador correspondiente al flujo lanzado, de esta forma los comandos a ejecutar serían los siguientes:

1. Consultar el fichero ***log-error***:

```
1 Sintaxis
2 cd /tmp/fishubulogs/log-error-{id-stream}
```

2. Consultar el fichero ***log***:

```
1 Sintaxis
2 cd /tmp/fishubulogs/log-{id-stream}
```

Otros errores serán almacenados en la carpeta ***logs*** de cada máquina *docker* ejecutada. Para poder observar estos errores será necesario adentrarse en el contendido del que se quiera obtener información, esta acción se realiza mediante alguno de los siguientes comandos:

```
1 Sintaxis
2     docker exec -it <container_name> bash}
```

```
1 Sintaxis
2     docker exec -it <container_id> bash}
```

## Fallos de memoria

### Fallos en la caída del flujo

Parte del fallo redactado por Jose Luis Garrido Labrador y ampliado por Lucía Núñez Calvo

Si la caída del flujo deja una excepción del tipo ***Caused by: java.io.EOFException***, como se puede observar en la figura D.1, o se observa el siguiente mensaje de error dentro del script /tmp/fishubulogs/log-error-id-stream ***StreamingContext: StreamingContext has already been stopped*** como se puede apreciar en la figura D.2, implica que los datos que debe procesar el flujo son mayores que los que caben en memoria. Para solucionar este problema será necesario dar más memoria a cada *worker*. El valor recomendado es de 2 GiB por cada núcleo de *worker* pero se puede aumentar si el usuario lo considera oportuno.

```
Python worker exited unexpectedly (crashed)
    at org.apache.spark.api.python.BasePythonRunner$ReaderIterator$$anonfun$3.applyOrElse(PythonRunner.scala:490)
    at org.apache.spark.api.python.BasePythonRunner$ReaderIterator$$anonfun$3.applyOrElse(PythonRunner.scala:479)
    at scala.runtime.AbstractPartialFunction.apply(AbstractPartialFunction.scala:36)
    at org.apache.spark.api.python.PythonRunner$$anon$1.read(PythonRunner.scala:597)
    at org.apache.spark.api.python.PythonRunner$$anon$1.read(PythonRunner.scala:575)
    at org.apache.spark.api.python.BasePythonRunner$ReaderIterator.hasNext(PythonRunner.scala:410)
    at org.apache.spark.InterruptibleIterator.hasNext(InterruptibleIterator.scala:37)
    at scala.collection.Iterator$class.foreach(Iterator.scala:891)
    at org.apache.spark.InterruptibleIterator.foreach(InterruptibleIterator.scala:28)
    at org.apache.spark.api.python.PythonRDD$.writeIteratorToStream(PythonRDD.scala:224)
    at org.apache.spark.api.python.PythonRunner$$anon$2.writeIteratorToStream(PythonRunner.scala:561)
    at org.apache.spark.api.python.BasePythonRunner$WriterThread$$anonfun$run$1.apply(PythonRunner.scala:346)
    at org.apache.spark.util.Utils$.logUncaughtExceptions(Utils.scala:1945)
    at org.apache.spark.api.python.BasePythonRunner$WriterThread.run(PythonRunner.scala:195)
Caused by: java.io.EOFException
    at java.base/java.io.DataInputStream.readInt(DataInputStream.java:397)
    at org.apache.spark.api.python.PythonRunner$$anon$1.read(PythonRunner.scala:582)
    ... 10 more
```

Figura D.1: Error de tipo *Caused by: java.io.EOFException*.

```

at java.base/java.net.Socket.<init>(Socket.java:454)
at java.base/java.net.Socket.<init>(Socket.java:264)
at java.base/javax.net.DefaultSocketFactory.createSocket(SocketFactory.java:277)
at py4j.CallbackConnection.start(CallbackConnection.java:226)
at py4j.CallbackClient.getConnection(CallbackClient.java:238)
at py4j.CallbackClient.getConnectionLock(CallbackClient.java:250)
... 25 more
22/06/04 19:55:39  WARN StreamingContext: StreamingContext has already been stopped
22/06/04 19:55:41  WARN StreamingContext: StreamingContext has already been stopped
22/06/04 19:55:41  WARN StreamingContext: StreamingContext has already been stopped
22/06/04 19:55:41  WARN StreamingContext: StreamingContext has already been stopped
22/06/04 19:55:41  WARN StreamingContext: StreamingContext has already been stopped

```

Figura D.2: Error de tipo *StreamingContext: StreamingContext has already been stop-ped*.

### Fallos con *Docker*

En caso de que se proceda al lanzamiento de los contenedores y la salida por pantalla sea ***The server is not running***, como se observa en la figura D.3, el problema puede ser que se haya agotado la memoria en el directorio `/var/cache/apt/archives/`. Para solventar este problema se tendrá que ejecutar siguiente el comando:

```

1 Sintaxis
2 docker system prune --all

```

Tras su ejecución se deberán restaurar las imágenes (orden build).

```

(base) lnc1002@gamma-gpu:~/TFM4/src/scripts/deploy$ ./start-server $(pwd)/output55 2 6 2 8g
50e5d852cac1b8a6ffdd7cffdc3ec562d048b3480b18962470b7edc7b52b734a
71e5155b96524b90465258da76d85dc4b0d988eab803b33920ffcc0f99e78d3cb
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
32b6cb6d5505a2e1bf961ad469ec31d7ac45f8ee870b458d5c39038f46dba96
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
352018fb17a8b1cf7801cd633a52798fc35ef8e6522dd094351ff0e15d679d23
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
b23b2e4cbe91022651d3eea6efcb692ce3c2f6dfc799d0546729d906603085fb
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
a5fc1620a0e689a0a67ae11267b6289d72e15a26298d3dd1bd088d07818d8d6b9
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
22dd01039f77b85793c1657709fdc1eac13e89f670f4a72b2cfdc4030c28af08
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
412bf0335aaedbbfb57cfaba152f109952360be2fe3210d444840ea4fdb58304
Recreating kafka_zookeeper_1 ...
Recreating kafka_zookeeper_1 ... done
Recreating kafka_kafka_1 ...
Recreating kafka_kafka_1 ... done
(base) lnc1002@gamma-gpu:~/TFM4/src/scripts/deploy$ ./new-stream "-f 5 --resize=1/2" "-f 5 -a -p 15 -b -c"
The server is not running

```

Figura D.3: Error de tipo *The server is not running*.

## Error de permisos

Si el proyecto se está ejecutando sobre la máquina *gamma* del grupo *ADMIRABLE* de la Universidad de Burgos y tras intentar lanzar el servidor aparecen numerosos mensajes de tipo ***Get permission denied while trying to connect to the Docker daemon socket at unix***, como el que se puede apreciar en la imagen D.4, quiere decir que el usuario no está presente en el grupo *docker*.

```
(root) [root@gamma ~]# /opt/PyTorch-2F-matrix/vision/extraOpt.py$ ./start-server $(pwd)/output 2 & 1 2g
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/networks/create: dial unix /var/run/docker.sock: connect: permission denied
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/build/buildargs=&7B%7D&cach
eFrom=&5$8NSD&groupParent=&cpuPeriod=&cpuQuota=&cpusetCpus=&cpusetMems=&cpusShares=&dockerfile=&labels=&7B%7D&memory=&8memswap=&networkMode=&default&t=rm=&session=tie4
9nef7w8ltrziihlh2qbashmsize=&dt=&spark-base=fishX3A2.4.&target=&ulimits=null&version=1: dial unix /var/run/docker.sock: connect: permission denied
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/build/buildargs=&7B%7D&cach
eFrom=&5$8NSD&groupParent=&cpuPeriod=&cpuQuota=&cpusetCpus=&cpusetMems=&cpusShares=&dockerfile=&labels=&7B%7D&memory=&8memswap=&networkMode=&default&t=rm=&session=rwH
o2g23ilceyus2qfs1c&shmsize=&dt=&spark-master=fishX3A2.4.&target=&ulimits=null&version=1: dial unix /var/run/docker.sock: connect: permission denied
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/build/buildargs=&7B%7D&cach
eFrom=&5$8NSD&groupParent=&cpuPeriod=&cpuQuota=&cpusetCpus=&cpusetMems=&cpusShares=&dockerfile=&labels=&7B%7D&memory=&8memswap=&networkMode=&default&t=rm=&session=rwH
2leiw3td4x0erf3nxb0y&shmsize=&dt=&spark-worker=fishX3A2.4.&target=&ulimits=null&version=1: dial unix /var/run/docker.sock: connect: permission denied
[ERRNO 2000] failed to dial gRPC: cannot connect to the Docker daemon. Is 'docker daemon' running on this host?: dial unix /var/run/docker.sock: connect: permission denied
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/containers/crea
t?name=spark-worker-fishhub-1: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/containers/crea
t?name=spark-worker-fishhub-1: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/containers/crea
t?name=spark-worker-fishhub-2: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/containers/crea
t?name=spark-worker-fishhub-3: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/containers/crea
t?name=spark-worker-fishhub-4: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/containers/crea
t?name=spark-worker-fishhub-5: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://>2Fvar%2Frun%2Fdocker.sock/v1.40/containers/crea
t?name=spark-worker-fishhub-6: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
[ERRNO: Couldn't connect to Docker daemon at http+docker://localunixsocket - is it running?
If it's at a non-standard location, specify the URL with the DOCKER_HOST environment variable.
```

Figura D.4: Error de tipo *Get permission denied while trying to connect to the Docker daemon socket at unix*.

Para solventar este problema, un administrador del sistema deberá ejecutar el siguiente comando:

1	Sintaxis
2	usermod -a -G docker nombre-usuario

## Los fotogramas no se procesan

Parte del fallo redactado por Jose Luis Garrido Labrador y ampliado por Lucía Núñez Calvo

Si los fotogramas no se procesan y observa la existencia del fichero de *log/notprocesslog* en los *workers*, significa que ha existido un error mientras se procesaba el fotograma. En el mismo *log* se encuentra el origen del fallo y probablemente se deberá a que el fichero de *extraOpt.py* incluye información errónea o una ruta incorrecta.

Para solucionar esto se debe volver a cargar el fichero `extraopt.py` y por tanto se han de volver a construir las imágenes (*orden build*) desde la `fishhubu-env`. En el caso de que el error persista, usar el flag `-no-cache` a la hora de reconstruir las imágenes.

## Error de *Kafka*

Algunos de los errores que pueden aparecer ante un usuario impaciente son los siguientes, como se puede apreciar en la imagen D.5:

1. *Error while executing topic command : Replication factor: 1 larger than available brokers: 0. ERROR org.apache.kafka.common.errors.InvalidReplicationFactorException: Replication factor: 1 larger than available brokers: 0. (kafka.admin.TopicCommand)*
  2. *WARN [AdminClient clientId=adminclient-1] Connection to node -1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available. (org.apache.kafka.clients.NetworkClient)*

Figura D.5: Error de tipo *kafka*.

Estos errores únicamente le informan al usuario de que no ha dejado suficiente tiempo para que se cargue *Kafka* correctamente. El tiempo de carga puede variar por múltiples factores, por esa razón puede, que tras lanzar el servidor se intente crear un flujo y no nos muestre el error, mientras que en otras ocasiones si lo haga. Por ello, es recomendable dejar trascurrir unos segundos desde que se lanza el servidor.

En caso de que aparezca este error, únicamente se pararía el flujo creado y se volverían a lanzar los contenedores.

## Erros de versión de *Nvidia*

Redactado por Jose Luis Garrido Labrador

Si a la hora de ejecutar el flujo, en el arranque ocurre una excepción del tipo ***forward compatibility was attempted*** indica que la versión

instalada de *Nvidia* sobre *docker* no es compatible con la versión instalada en el equipo.

Para solucionarlo es necesario cambiar el fichero *Dockerfile* de la carpeta *src/dockers/fishubu/base* y cambiar la versión que se instala por la que se tiene en el equipo. Es necesario que al menos sea la versión 440.

### Error con la conexión a *Nvidia*

[Redactado por Jose Luis Garrido Labrador](#)

Si el flujo al iniciar da el error ***could not select device driver*** significa que la extensión de *docker* para la compatibilidad con *Nvidia* no está instalada o no se ha reiniciado el servicio de *docker* desde que se instaló.

Se soluciona instalando la extensión de *docker* «*docker-nvidia*» y reini-ciando el servicio.

## D.6. Pruebas del sistema

Como se ha podido observar, el proyecto se ha dividido en dos fases principales. Una primera fase fue el análisis de librerías y la segunda fase fue condensar todo lo investigado en la implementación del objetivo del proyecto.

Prácticamente todas las pruebas se han corrido sobre distintos *notebooks* ya que esta herramienta permite organizar de una manera muy intuitiva cómoda y visual los resultados obtenidos, pero ¿qué se entiende por pruebas?.

Un aspecto a tener en cuenta es la dificultad de ejecutar pruebas unitarias en minería de datos. Para ejecutar una prueba se debe saber de manera previa el resultado, en nuestro caso este resultado sólo se consigue mediante un análisis visual por parte del usuario. Es el propio programador el que tiene que comprobar si la secuencia obtenidas se corresponde con la secuencia esperada. Este resultado se ha podido comprobar de dos maneras:

1. **Pruebas mediante código:** el procedimiento a seguir para realizar este tipo de pruebas es muy sencillo. Tras analizar el vídeo que contiene la secuencia de ejercicios completos, localizamos visualmente el vídeo que queremos encontrar y nos quedamos con el instante de tiempo en el que comienza y finaliza. El siguiente paso es una mera regla de tres. Si se tienen un total de *X frames* y el vídeo cuenta con una duración de *Z* instantes de tiempo, el momento justo en el que empieza o finaliza

el ejercicio concreto encontrado dentro del vídeo que contiene todos los ejercicios será igual a

$$\frac{frame\_encontrado * Z}{X} \quad (\text{D.1})$$

2. **Pruebas mediante la aplicación de escritorio:** el procedimiento a realizar en este caso es más sencillo porque el cálculo anterior ya se encuentra implementado en la aplicación. En este caso será necesario cargar tanto el vídeo concreto como el vídeo que comprende toda la secuencia de ejercicios y proceder a recortar el vídeo. Visualizando el recorte en la aplicación se podrá comprobar lo bien o mal que se está detectando la secuencia.

Finalmente comentar que las pruebas han sido esenciales en este proyecto para poder seleccionar el mejor algoritmo para el objetivo perseguido, es decir, el algoritmo que consiga localizar una amplia colección de secuencias en un tiempo relativamente aceptable.



## *Apéndice E*

---

# **Documentación de usuario**

---

## **E.1. Introducción**

En este apartado se detallarán las principales funcionalidades que ofrece la aplicación de escritorio a si como todo lo necesario para que los usuarios puedan instalar y ejecutar la aplicación fácilmente.

## **E.2. Requisitos de usuarios**

Para poder correr la aplicación será necesario tener instalado *Python*, al menos la versión 3.8.8 e instalar los requerimientos especificados en el fichero *app/requirements.txt*. Esta aplicación está pensada para ser ejecutada sobre el sistema operativo *Windows*.

## **E.3. Instalación**

Los pasos básicos para instalar el proyecto y con ende la aplicación son los siguientes:

1. Clonar repositorio e ir al directorio *app/*.
2. Comprobar que se encuentran almacenados algunos vídeos en las diferentes carpetas que contiene el directorio *app/*.
3. Ejecutar el comando:

```

1 Sintaxis
2     install.cmd

```

o en su defecto

```

1 Sintaxis
2     pip install -r requirements.txt

```

Finalmente para ejecutar la aplicación: Para ejecutar la aplicación:

```

1 Sintaxis
2     python main.py

```

## E.4. Manual del usuario

### Cambiar el modo de la aplicación

Para cambiar el modo de la aplicación observar figuras E.1 E.2. Al cambiar de modo se ejecutarán las siguientes acciones:

1. **Cambio de colores:** si el usuario se encuentra en *modo ejemplo ON* aparecerá una ventana con tonos verdes azulados y azules, mientras que si el usuario se encuentra en *modo ejemplo OFF* los tonos serán de color lila. Al cambiar el color de toda la aplicación dependiendo del modo de ejecución en el que se encuentre el usuario se logra evitar confusiones sobre en que modo se encuentra.
2. **Cambio del texto del botón:** es imprescindible que el usuario sepa en todo momento en que modo de ejecución se encuentra por lo que el cambio de colores no es suficiente, mediante el botón *ON/OFF* se informará del modo en el que se encuentra la aplicación.
3. **Bloqueo o desbloqueo de casillas:** las casillas por las que se inserta el *frame* de inicio y final aparecerán bloqueadas si el usuario se encuentra en *modo ejemplo ON* ya que no necesitará hacer uso de ellas.

4. **Actualización de vídeos:** si la aplicación cambia de modo, se actualizarán los vídeos mostrados en las listas de selección. Cada modo puede contener diferentes vídeos.
5. **Descarte de vídeos:** si la aplicación está reproduciendo un vídeo y se ejecuta un cambio de modo, automáticamente descartará el vídeo.



Figura E.1: Aplicación en modo ejemplo ON.



Figura E.2: Aplicación en modo ejemplo OFF.

## Mostrar y borrar un vídeo

Una vez se arranca la aplicación de escritorio, los vídeos son cargados automáticamente por la aplicación y mostrados en la barra de selección.

### Mostrar un vídeo

Para mostrar un vídeo ver figura E.3. Al mostrar un vídeo se ejecutarán las siguientes acciones:

1. **Seleccionar un vídeo:** lo primero que se deberá hacer es escoger un vídeo de los mostrados en la lista de selección.
2. *Pinchar en el botón **Mostrar**:* tras seleccionar un vídeo y seguidamente pinchar en este botón, la aplicación cargará el vídeo seleccionado en su ventana correspondiente y empezará con su reproducción.
3. **Descartar vídeos previos:** si se desea cargar un vídeo cuando ya hay otro en ejecución, es muy importante descartar el vídeo previo. En caso de no realizarse correctamente esta acción, el vídeo seleccionado no se cargará.



Figura E.3: Carga y descarte de vídeos.

### Descartar un vídeo

Para descartar un vídeo ver figura E.3. Al descartar un vídeo se ejecutarán las siguientes acciones:

1. **Seleccionar el vídeo:** para que el vídeo se descarte correctamente se deberá seleccionar el vídeo que está en ejecución y que se desea eliminar.
2. **Pinchar en el botón *Descartar*:** una vez seleccionado el vídeo se pulsará este botón. Esta acción simplemente deja de reproducirlo en su ventana correspondiente, pero no elimina el vídeo de la lista de selección, por esta razón se pueden cargar y descartar tantas veces como se desee.
3. **EL vídeo se bloqueará:** como se ha comentado anteriormente los vídeos no son eliminados por completo si no que únicamente se bloquean y dejan de reproducirse. Una vez el vídeo está bloqueado se podrán añadir nuevos vídeos.

### Interaccionar con el vídeo

Una vez esté cargado un vídeo, se podrá pausar y reanudar tantas veces como el usuario desee.

### Reproducir un vídeo

Para reproducir un vídeo se tendrán en cuenta estas sencillas acciones:

1. **Pinchar en el botón *play*:** los vídeos pausados o detenidos únicamente empezarán a reproducirse pulsando este botón.
2. **EL vídeo empezará a reproducirse:** si el vídeo se encuentra pausado o detenido, ya sea porque el usuario ha presionado el botón *stop* o porque la duración del vídeo ha llegado a su fin, una vez se pincha sobre el botón *play*, empezará su reproducción.

## Pausar un vídeo

Para pausar un vídeo se tendrán en cuenta estas sencillas acciones:

1. **Pinchar en el botón *pause*:** este botón únicamente permitirá la pausa del vídeo en reproducción. No ejecutará el vídeo de nuevo al hacer doble *click*.
2. **EL vídeo se pausará:** el vídeo permanecerá pausado hasta que se ejecute una acción de reproducción.

## Detener un vídeo

Para detener un vídeo se tendrán en cuenta estas sencillas acciones:

1. **Pinchar en el botón *stop*:** este botón únicamente permitirá la detención inmediata del vídeo en reproducción. No ejecutará el vídeo de nuevo al hacer doble *click*.
2. **EL vídeo se parará:** el vídeo permanecerá detenido hasta que se ejecute una acción de reproducción. En este caso, el vídeo comenzará desde el instante inicial.

## Recortar un vídeo

Los vídeos se podrán recortar en ambos modos de ejecución. En las imágenes E.4 y E.6 se puede observar como el ejercicio realizado por el vídeo que simula al terapeuta es localizado en el vídeo que simula al paciente y posteriormente muestra dicho resultado.

### Recortar un vídeo en modo ejemplo ON

Para recortar un vídeo en *modo ejemplo ON* ver figura E.4 y tener en cuenta las siguientes acciones:

1. **Seleccionar un vídeo concreto:** se deberá escoger el vídeo que se desea encontrar dentro de la secuencia de ejercicios.
2. **Seleccionar un vídeo completo:** también se deberá escoger el vídeo del que se desea obtener el recorte.
3. **Pinchar en el botón *Recortar*:** para ver el resultado final del recorte se deberá pulsar en este botón. En cuanto el vídeo recortado sea cargado empezará a reproducirse.



Figura E.4: Recorte de vídeos en modo ejemplo ON.

#### Recortar un vídeo en modo ejemplo OFF

Para recortar un vídeo en *modo ejemplo OFF* ver figura E.6 y tener en cuenta las siguientes acciones:

1. **Seleccionar un vídeo concreto.**
2. **Seleccionar un vídeo completo.**
3. **Indicar el *frame* de inicio:** para el generar el recorte será necesario indicarle el punto de inicio en la casilla correspondiente.
4. **Indicar el *frame* de final:** también será necesario especificarle el punto en el que se desea que finalice el resultado.
5. **Pinchar en el botón *Recortar*:** una vez se tienen ambos valores cargados, tras seleccionar este botón aparecerá el vídeo resultado reproduciéndose automáticamente.

#### Excepciones e información de la aplicación

Finalmente comentar los siguientes aspectos relevantes e la aplicación:



Figura E.5: Recorte de vídeos en modo ejemplo OFF.

- Si el usuario se encuentra en el *modo ejemplo ON* y selecciona el botón recortar sin haber especificado los *frames* de inicio y de final, mostrará el siguiente aviso:

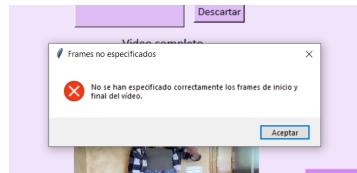


Figura E.6: Aviso de falta de frames.

- Si se selecciona el desplegable **Ayuda** se mostrarán las opciones *Proyecto* y *Modo ejemplo* que mostrarán la siguiente información:

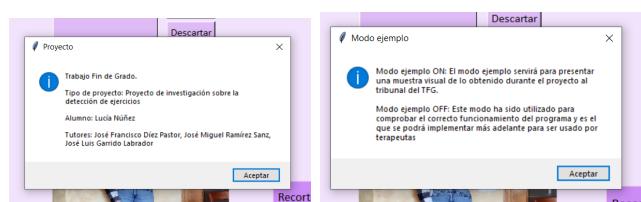


Figura E.7: Detección de animales usando diferentes algoritmos.

---

# Bibliografía

---

- [1] Diagrams.net. <https://app.diagrams.net>, 29 November 2021.
- [2] ArchLinux Wiki Authors. Docker. <https://wiki.archlinux.org/index.php?title=Docker&oldid=622596>, 2020. [Internet; descargado 31-junio-2020].
- [3] Indeed. Salarios para empleos de A.C.S. informáticos en España. <https://www.indeed.es/cmp/A.c.s.-Inform%C3%A1ticos/salaries>, jun 2019.
- [4] NVIDIA. Nvidia software. <https://www.nvidia.com/es-la/drivers/nvidia-software/>, 2021.
- [5] NVIDIA. Zona cuda. <https://developer.nvidia.com/cuda-zone>, 2022.
- [6] Pythonista. Anaconda python y conda. <https://elpythonista.com/anaconda>, 2022.
- [7] Tensorflow. Plataforma de extremo a extremo de código abierto para el aprendizaje automático. <https://www.tensorflow.org>, 2022.
- [8] Manuel Trigás Gallego. Metodología scrum. 2012.
- [9] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.