UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Identificación de Parkinson por visión artificial

Documentación Técnica

Presentado por Catalin Andrei Cacuci
en Universidad de Burgos — 25 de marzo de 2023
Tutor: Álvar Arnaiz González

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

A.2. Planificación temporal

La planificación del proyecto se llevará a cabo mediante una metodología ágil, en concreto Scrum, con el apoyo de la herramienta ZenHub para la gestión de tareas y sprints dentro de GitHub. En este caso cada sprint tiene una duración de una semana.

Preparación

Antes de comenzar el sprint 1 se realizaron las siguientes tareas:

- Organizar el proyecto en diferentes carpetas con las partes que lo componen.
- Implementar la conectividad básica entre los contenedores de Docker que implementan la aplicación web.
- Implementar el parseado de los nombres de archivo de los vídeos.
- Implementar la extracción de puntos del esqueleto de la mano mediante Mediapipe Hands.
- Crear un paquete instalable o librería de Python con las utilidades que se utilizan en la fase de investigación para facilitar su uso posterior en la aplicación web.

Sprint 1

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

Apéndice B

Especificación de Requisitos

B.1. Introducción

B.2. Objetivos generales

B.3. Catalogo de requisitos

B.4. Especificación de requisitos

Apéndice C

Especificación de diseño

C.1. Introducción

C.2. Diseño de datos

C.3. Diseño procedimental

C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

D.1. Introducción

Esta sección contiene toda la información que una persona externa debería tener para poder trabajar con las diferentes partes de este projecto.

D.2. Estructura de directorios

Los directorios del proyecto siguen la siguiente estructura:

- **app**: Contiene todo lo relacionado con los contenedores de Docker que conforman la aplicación web.
    - **api**: Contiene el código fuente de la API que implementa el framework FastAPI de Python.
    - **web**: Contiene el código fuente de la página web, se trata de una implementación SvelteKit, que es un framework de JavaScript.
    - **proxy**: Contiene la configuración de Nginx para el proxy inverso que se utiliza para acceder a la API y a la web desde el exterior.
- **docs**: Contiene el código fuente en LaTeXde esta documentación.
- **paddel**: Contiene el código fuente de la librería PADDEL, que tiene aquellas utilidades empleadas en la fase de investigación para facilitar su uso dentro de la aplicación web. Se trata de un paquete de Python que se puede instalar mediante el comando `pip`.

D.3. Manual del programador

Este proyecto utiliza la utilidad GNU Make para guardar secuencias de comandos y gestionar las dependencias que existen entre estas. Para esto se utiliza un archivo denominado *Makefile* en los directorios relevantes. En caso de que el sistema utilizado no disponga de la herramienta Make, se pueden simplemente consultar el archivo *Makefile* para ver los comandos que deberían ser utilizados.

PADDEL

La librería PADDEL dispone de un archivo *pyproject.toml* en el que se detalla la forma de instalación, dependencias y algunas configuraciones de herramientas de desarrollo.

Se incluye, además, un archivo *Makefile* que define los siguientes comandos:

- `make install-dependencies`: Instala la librería con las dependencias de desarrollo, utilizadas para acciones como formateado (*black*) o comprobación de tipos (*mypy*).
- `make lint`: Realiza varias comprobaciones sobre el código fuente para detectar problemas de formateado o tipado e informa al usuario en caso de que existan.
- `make format`: Realiza un formateado sobre el código fuente, eliminando sentencias `import` no utilizadas, reordenando estas sentencias `import` y formateando el código para darle un aspecto homogéneo.

Para el entorno de desarrollo es muy recomendable utilizar un entorno virtual para evitar realizar cambios sobre la instalación de Python a nivel de sistema. Existen varias herramientas que permiten realizar esto, como *venv* o *conda*. Con instalaciones que no se van a detallar en este documento al existir documentación extensa sobre el uso de ambas herramientas. Es importante destacar que la librería se ha creado en la versión 3.10 de Python, por lo que es recomendable utilizar esta versión para el entorno virtual.

Una vez creado el entorno virtual de Python, este se deberá activar de forma acorde a la herramienta utilizada y, a continuación, se puede ejecutar el comando `make install-dependencies`, en ausencia de la utilidad Make se puede utilizar `pip install -e .[dev]`, que instala el paquete en modo editable junto con las dependencias de desarrollo.

Con todo esto se puede utilizar el comando `jupyter notebook notebooks` para lanzar lanzar una instancia del kernel IPython en la carpeta *notebooks* donde se encuentra el código utilizado durante la fase de investigación, preprocesado y entrenamiendo de modelos.

Para ejecutar los notebooks de Jupyter se debe crear una carpeta `/data/raw` en la raíz del repositorio donde introducir los vídeos con el formato de nombre adecuado.

Aplicación

Los componentes de la aplicación web se encuentran en el directorio `app` se trata de tres contenedores de Docker, api, web y proxy. Para lanzar los contenedores en modo desarrollo, es decir, que se detecten los cambios en la mayoría de ficheros fuente y se actualizen los contenedores acordemente, se puede utilizar `make docker-up-dev`, siempre que el motor de Docker esté inicializado.

Para parar los contenedores se utiliza `make docker-down`, si, además, se desea liberar memoria y eliminar las imágenes creadas se utiliza `make docker-clean`.

La edición de ficheros fuente se puede realizar diréctamente desde el sistema anfitrión, pero, si se desea de ayudas como autocompletado y recomendaciones será necesario utilizar un entorno de desarrollo que pueda interactuar con contenedores, como por ejemplo *Visual Studio Code* con la extensión *Dev Containers*.

D.4. Compilación, instalación y ejecución del proyecto

Para desplegar el proyecto simplemente se debe ejecutar `make docker-up` desde la carpeta `app` con el motor de Docker inicializado, lo que inicializará la aplicación web en el equipo anfitrión.

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Bibliografía