UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Sistema de Etiquetado de Fitolitos Online Documentación Técnica

Presentado por Marta Monje Blanco
en Universidad de Burgos — 17 de septiembre de 2018
Tutores: Dr. Álvar Arnaiz González y Dr. José Francisco Díez Pastor

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

A lo largo del desarrollo del proyecto, vamos a trabajar bajo el método de *Scrum*, el cual propone un marco de trabajo dentro del desarrollo ágil[1] en que las entregas son incrementales e iterativas. Se realizarán *sprints*[2] semanales en los que se harán reuniones con los tutores para evaluar lo realizado a lo largo del *sprint* y planificar la continuación del siguiente. Para manejar el proyecto bajo esta filosofía de trabajo y facilitar su gestión, se usará a lo largo del proyecto un repositorio de *GitHub*: https://github.com/mmb0093/TFG_Fitolitos

A.2. Planificación temporal

Sprint 1

En este sprint se han cubierto las primeras tareas del proyecto. En este caso las tareas han estado relacionadas principalmente con la lectura y comprensión de la documentación previa del mismo trabajo. También se comenzó con la búsqueda de etiquetadores, para encontrar el más adecuado para el uso que le queremos dar.

[1] Página del Manifiesto ágil http://agilemanifesto.org/

[2] Iteración de tiempo prefijado en el cual se entrega un incremento del producto.

Sprint 2

En este segundo sprint la actividad ha estado centrada en la elección del etiquetador y en el aprendizaje del mismo. Finalmente se tomó la decisión de utilizar *Alp's Labeling Tool (ALT)*.

Se ha generado la documentación del manual de usuario para la instalación y uso del etiquetador y se ha creado una máquina virtual para facilitar el uso de la misma, ya que evita el tener que realizar la instalación, la cual puede ser algo confusa.

Sprint 3

Se ha realizado un script para comprobar la corrección de las coordenadas sobre las imágenes. El anterior etiquetador no etiquetaba bien y por lo tanto no se van a poder emplear las imágenes previamente etiquetadas.

Sprint 4

Se ha comenzado a investigar para realizar el despliegue de la aplicación. En un inicio se pensó en Heroku como servidor y Flask como framework y se han realizado pruebas para comprobar si es lo más adecuado.

Sprint 5

Se ha decidido cambiar el despliegue de Heroku a Digital Ocean con el uso de *Nanobox*. En cuanto a los casos de uso del sistema, de momento, se va a usar la estructura empleada en la versión previa del proyecto.

Sprint 6

En este sprint se ha realizado el login de la aplicación con Google y el uso de 'Flask-Dance'. Se pensó que sería buena idea, desde el punto de vista de la gestión de imágenes, hacer el login desde Dropbox, pero se descartó la idea durante la reunión y se decidió cambiarlo de nuevo a Google.

Sprint 7

Durante el despliegue se han dado algunos errores que han estado relacionados con el fichero *boxfile.yml*. Por lo general, en este fichero hemos de indicarle ciertos aspectos de la configuración que ha de seguirse para ubicar la aplicación funcional dentro del servidor. Esto es tarea de otros dos ficheros

que existen dentro de la aplicación llamados *gunicorn.py* y *nginx.config*, ambos necesarios. El problema reside en que durante el despliegue, el fichero nginx no parece tener los permisos necesarios para hacer efectivo dicho despliegue.

Sprint 8

Para este srprint se han planteado tareas que conciernen al etiquetador.

Se ha introducido la fila de botones del etiquetador, las cuales resuelven la problemática de tener que escribirlos a mano cada vez que etiquetas algún elemento.

En total se añadieron 7 botones con lo tipos de fitolitos definidos y se retocó dicho código para poder darles la funcionalidad que se deseaba.

Análogamente se continuó trabajando con los errores del despliegue que se produjeron durante el anterior sprint.

Sprint 9

Durante este sprint se continuó con las funcionalidades de los botones del etiquetador.

Ya eran funcionales pero se quería que las etiquetas fuesen redimensionables y que se pudiesen mover. Se tomó la decisión de que hubiese un tipo de fitolito por defecto en el caso de que e usuario no hubiese seleccionado ninguna etiqueta antes de ponerse a dibujar etiquetas en las imágenes.

Se añadió la opción de guardado mediante la cual sepueden guardar las imágenes con las que se ha trabajado, en el servidor. Debido a los problemas que hubo con el despliegue, el cual no me dejaba acceder al almacenaje en el servidor debido a la falta de permisos, se decidió mantener la opción y trabajar en local hasta solucionar este problema.

Sprint 10

Durante este sprint se realizaron tareas de refactorización del código. Muchas de las funcionalidades de los botones de etiquetador, así como parte del sistema de generado de *csv* con las etiquetas, habían dejado el código ilegible, así que se eliminaron las funciones que ya no eran necesarias, variables que no servían y código duplicado para que el mantenimiento del código se simplificase tanto como se pudiera.

También se etiquetó un conjunto de imágenes de prueba para empezar a tener un conjunto de entrenamiento y poder usarlo cuando estuviese el clasificador.

Por otra parte se empezó a investigar como se iba a hacer el clasificador, con que librerías y que modelos de entrenamiento. Complementario a esto, se buscó mucha información sobre los conceptos teóricos que concernían al entorno del reconocimiento y la inteligencia artificial(AI).

Sprint 11

En este sprint se redefinió la estructura de carpetas del proyecto para hacerla más comprensible, eficiente y fácil de usar.

Se planteó y estudió la idea de meter un perfil para el administrador que va a estar al otro lad de la aplicación.

Por otro lado se eliminó el despliegue realizado en *Nanobox* [3] y se empezó la búsqueda de otras alternativas que funcionasen mejor. Finalmente se optó por desplegar con *Pythonanywhere* y funcionó perfectamente.

Sprint 12

Durante este sprint se modificó completamente la apariencia de la aplicación para darle un acabado más profesional.

Además se volvió a incluir la opción de salir del perfil de la aplicación, ya de forma definitiva. Se añadieron los nombres de los tutores y el mío propio dentro del pie de página y se añadió un enlace al repositorio en *GitHub*[4]

Por otra parte, se arregló, dentro del etiquetador, la funcionalidad de poder editar el tipo del fitolito, es decir, a qué clase pertenece. Se tomó la decisión de que si se desea cambiar el tipo del fitolito que se tuvieses marcado y volverlo a dibujarlo en la imagen.

Sprint 13 Última semana

En este sprint nos estamos dedicando casi por completo a la documentación.

Se va a intentar entrenar a un modelo por *notebooks* de *Jupiter* [5] y

[3]Página inicial de *Nanobox*: https://nanobox.io/

[4]Enlace para acceder al repositorio de la aplicación: https://github.com/mmb0093/TFG_Fitolitos

[5]Página de documentación https://ipython.org/notebook.html

mediante *Google colab* [6] para hacer pruebas con posibles etiquetadores.

Se remataron pequeños detalles de la interfaz de la página del etiquetador.

Gráficas globales

GitHub nos ofrece gráficos que representan de forma global la interacción con el repositorio durante el proyecto.

- ***Commits***: Esta gráfica muestra, mediante un grafo, la cantidad de commits que se han hecho a lo largo de los meses de trabajo.

  Durante los meses de verano a no hay cambios en el repositorio debido a que empecé a trabajar en una empresa a tiempo completo y no pude compaginarlo con el proyecto.

  ver imagen

Figura A.1: Grafo con la representación de los commits a los largo del proyecto

- ***Gráfica de elementos añadidos y elementos eliminados***

[6] Página de documentación de google https://colab.research.google.com/notebooks/welcome.ipynb

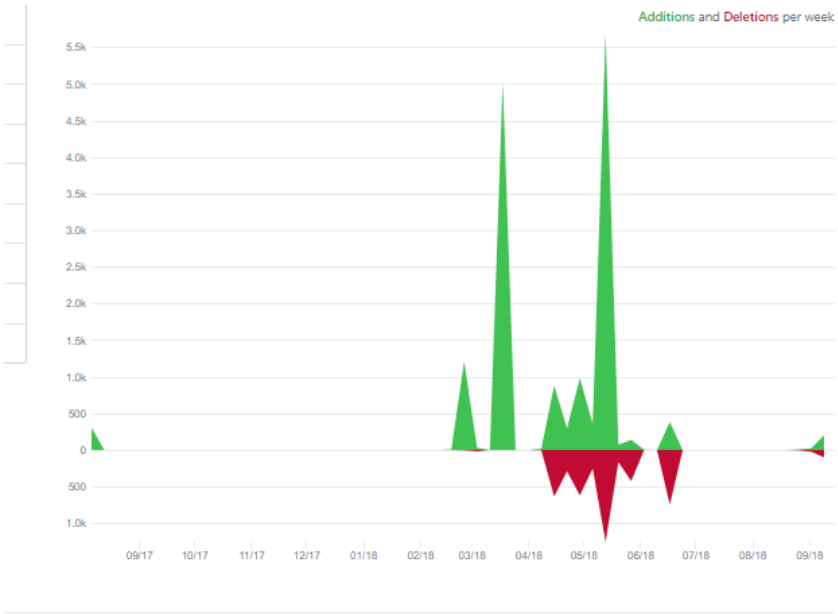

Figura A.2: Gráfica de elementos añadidos(verde) y para los eliminados (rojo) a lo largo del proyecto

En esta gráfica se muestra en verde la cantidad de objetos es añadida y lo rojo los elementos que se han borrado

A.3. Estudio de viabilidad

En este apartado se va a estudiar si es factible tanto a nivel económico como legal, llevar a cabo el proyecto dentro de un entorno empresarial.

Viabilidad económica

En esta sección se analizarán los costes económicos que se hubieran dado en el supuesto desarrollo del proyecto a nivel empresarial.

Para realizar este apartado se va a extraer la información necesaria de un artículo informativo escrito bajo la legislación actual vigente [**?**].

Costes materiales e informáticos

- *Hardware*: Para el desarrollo del proyecto, únicamente ha sido necesario un ordenador portátil valorado en 1250€ que se amortizará en 4 años, el cual, ya incluye todo lo necesario.
- *Software*: Actualmente los costes del software que se han utilizado han sido gratuitos debido a las licencias de estudiantes que ofrecen las empresas a las que pertenecen. Como estamos tratando con un entorno empresarial, tendremos que tener en cuenta los costes del software destinados para este fin. Se van a emplear el IDE de *JetBrains Pycharm*[7].

  En la tabla A.1 se pueden apreciar los costes y las amortizaciones[8] de ambas partes.

Costes Informáticos	**Coste total€**	**Amortización €**
Equipo(Ordenador portatil)	1250	208,32
JetBrains PyCharm	87,26	87,26
Total	1337,26	295,58

Tabla A.1: Costes Informáticos de *hardware* y *software*.

Costes de personal

Para este proyecto solo se ha necesitado un trabajador encargado del desarrollo, el cual ha trabajado a tiempo parcial.

Dicho empleado supondrá un coste para la empresa, el cual se calculará mediante la suma del salario bruto que este reciba, más la cotización de la seguridad social de la empresa.

Para calcular la parte de la seguridad social hay que tener en cuenta los siguientes parámetros:

- Contingencias comunes (empresa): 23,6 %

[7]Enlace a la página principal de *Pycharm*: https://www.jetbrains.com/pycharm/
[8]Amortizaciones en 8 meses, que es lo que está durando el proyecto.

- Desempleo de tipo general (empresa): 5,5 %
- Formación Profesional (empresa): 0,6 %
- Fondo de Garantía Salarial (FOGASA): 0,2 %

El total del porcentaje de la seguridad social por parte de la empresa es 29,9 %.

Por otra parte hay que distinguir los gastos del propio empleado, que de acuerdo con la legislación vigente son los siguientes:

- Contingencias comunes (trabajador): 4,7 %
- Desempleo de tipo general (trabajador): 1,55 %
- Formación profesional (trabajador): 0,1 %

La suma de los costes del propio empleado asciende a 6,35 %.

El total del porcentaje de la seguridad social es la suma de los dos porcentajes calculados, haciendo un total de 36,25 %.

Costes de personal	**Importe €**
Salario mensual bruto	1.500
Retención IRPF (15 %)	225
Seguridad social del empleado (6,35 %)	95,25
Salario mensual neto	1179,75
Salario total en 8 meses	9438
Seguridad social (Empresa) (29,9 %)	762
Coste total mensual	2821,96
Coste total	22.575,7

Tabla A.2: Costes de personal por parte de la empresa.

En la tabla A.2 se van a emplear los anteriores porcentajes para calcular los hipotéticos gastos ocasionados por el personal que conforma la empresa, como si se tratase de un caso real.

Librería	Versión	Licencia
Tensorflow	1.0	Apache 2.0
Numpy	1.10.2	BSD
Bootstrap	4.0.0	MIT
Flask	1.0	BSD
Flask Dance	1.0.0	MIT

Tabla A.4: Licencias de las librerías Usadas

Coste total: personal e informático

El coste a asumir en los 8 meses que dura el desarrollo, teniendo en cuenta al personal y el equipo informático, queda reflejado en la tabla A.3.

Coste total	Importe total €
Coste de personal	9438
Costes informáticos	295
Coste total	9733

Tabla A.3: Suma de costes de personal y costes informáticos al final del proyecto.

Viabilidad legal

Licencia del proyecto

Este proyecto está publicado bajo licencia *Apache 2.0* [**?**], que es actualmente una de las más usadas para proyectos porque es muy flexible y permisiva. Los derechos de autor de los proyectos amparados por esta licencia se deben de conservan tanto en el código fuente como en los archivos binarios.

Licencia de las librerías usadas

Se han usado varias librería a lo largo del proyecto, todas ellas en su versión más reciente, siendo todas de licencia libre. Se muestran todas en la tabla A.4.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este anexo se van a especificar y formalizar los objetivos y requisitos del proyecto mediante tablas y diagramas.

B.2. Objetivos generales

Los objetivos globales de este proyecto son:

- Confeccionar una herramienta que permita el etiquetado de distintos tipos de fitolitos.
- Ubicar dicha herramienta en un servidor web y añadir control de acceso.
- Que se puedan subir imágenes al servidor junto a la información que contengan las etiquetas para poder entrenar un modelo.

B.3. Catálogo de requisitos

En función de los objetivos que se han especificado, se va a extraer el conjunto de requisitos, funcionales y no funcionales, para el diseño de la aplicación.

Requisitos funcionales

Se van a recoger los requisitos funcionales [**?**], que son aquellos que definen algún tipo de función, dentro de la aplicación.

- **RF-1** Confeccionar una herramienta online que permita subir imágenes para ser etiquetadas.
  - **RF-1.1** Identificarse a través del login de *Google*
  - **RF-1.2** En el caso de que los datos introducidos sean erróneos, impedir el acceso a la aplicación.
  - **RF-1.3** En el caso de que los datos sean correctos, permitir el acceso a la aplicación.
  - **RF-1.4** Una vez dentro de la aplicación, permitir cerrar la sesión.
  - **RF-1.5** Una vez la sesión esté cerrada, que devuelva al usuario a la vista del login.
  - **RF-1.6** Escoger imágenes del equipo desde el que se esté trabajando.
  - **RF-1.7** Mostrar las imágenes en el etiquetador.
- **RF-2**: Gestionar las etiquetas que se dibujan sobre las imágenes.
  - **RF-2.1** Seleccionar el tipo de fitolito que se va a etiquetar.
  - **RF-2.2** Dibujar la etiqueta sobre la imagen.
  - **RF-2.3** Mover etiquetas previamente dibujadas.
  - **RF-2.4** Redimensionar etiquetas previamente dibujadas.
  - **RF-2.5** Borrar etiquetas previamente dibujadas.
- **RF-3** Que se puedan subir imágenes al servidor junto a la información que contengan las etiquetas para poder entrenar un modelo.
  - **RF-3.1** Guardar las imágenes subidas en el servidor.
  - **RF-3.2** Generar la información de las etiquetas en formato csv.
  - **RF-3.3** Guardar las etiquetas generadas en el servidor y descargarlas en local.
  - **RF-3.4** Mostrar las imágenes en la galería cuando se suban al servidor.
  - **RF-3.5** Eliminar del etiquetador aquellas imágenes que se hayan guardado.

Requisitos no funcionales

Se van a recoger los requisitos no funcionales [**?**], que son aquellos que especifican los *atributos de calidad* [**?**].

- **RNF-1** La aplicación ha de ser fácil de usar e intuitiva para el usuario.

B.4. Especificación de requisitos

En esta sección se va a especificar mediante diagramas de casos de uso en notación UML [**?**], los requisitos anteriormente definidos.

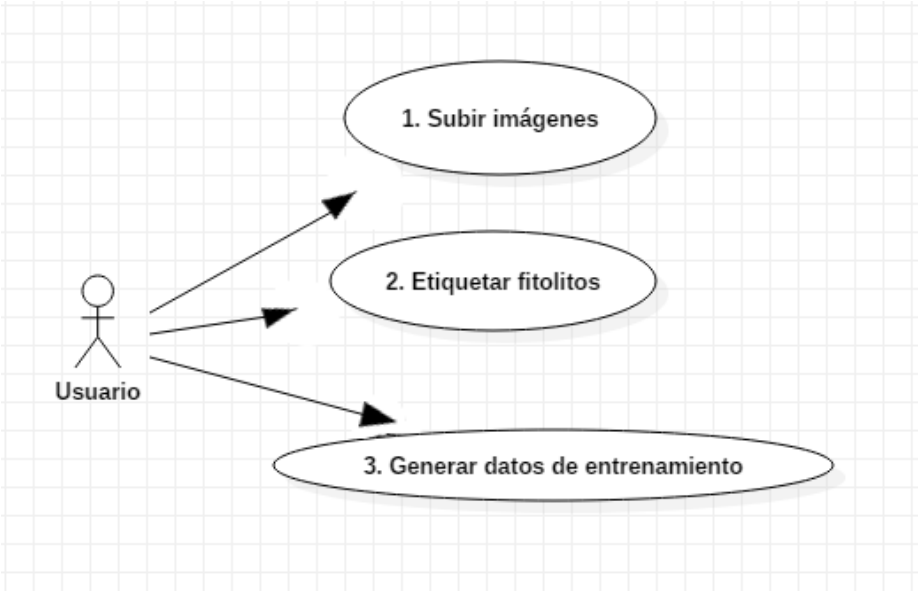

Figura B.1: Diagrama UML general de casos de uso de la aplicación.

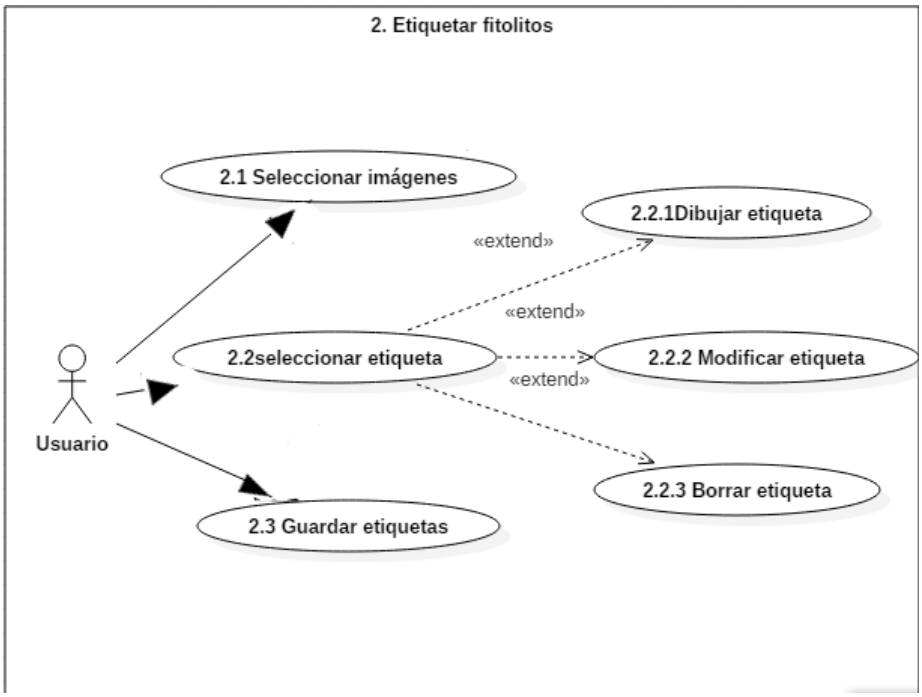

Figura B.2: Diagrama UML desglosado del caso de uso 2: *Etiquetar fitolitos*

En la imagen B.1 podemos ver el diagrama de casos general que vamos a tener. Las tablas que los formalizan son la B.1, que es del primer caso de uso, la B.2, correspondiente al segundo caso de uso y la B.3 que hará referencia al tercer caso de uso.

En la imagen B.2 podemos ver extendido el segundo caso de uso del primer diagrama. A este le corresponderán las tablas B.4 referente caso de uso 2.1, B.5 referente al caso de uso 2.2 y la B.6 a la que corresponde el caso de uso 2.3. Del caso de uso 2.2, extienden otros tres casos de uso: B.7 es la tabla de 2.2.1, la B.8 la del caso de uso 2.2.2 y la del caso de uso 2.2.3 B.9.

Caso de uso 1: Subir imágenes		
Descripción	El usuario puede subir imágenes que tenga en su equipo, puede seleccionar una o varias.	
Requisitos	RF-1	
	RF-1.1	
	RF-1.2	
	RF-1.3	
	RF-1.4	
	RF-1.5	
	RF-1.6	
	RF-1.7	
Precondiciones	Ninguna	
Secuencia normal	Paso	Acción
	1	El usuario pulsa el botón para subir imágenes.
	2	El usuario selecciona una imagen o conjunto de imágenes.
	3	El usuario selecciona abrir imágenes.
	4	Se cargan la imagen o las imágenes en el etiquetador.
Postcondiciones	La imagen se muestra en el canvas del etiquetador.	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	
Comentarios		

Tabla B.1: Caso de uso 1: Subir imágenes

Caso de uso 2: Etiquetado de fitolitos		
Descripción	El usuario puede dibujar, borrar y modificar etiquetas sobre las imágenes que ha subido	
Requisitos	RF-2	
	RF-2.1	
	RF-2.2	
	RF-2.3	
	RF-2.4	
	RF-2.5	
Precondiciones	Que haya imágenes subidas en el etiquetador	
Secuencia normal	Paso	Acción
	1	El usuario selecciona una imagen subida.
	2	El usuario selecciona un tipo de etiqueta
	3	El usuario dibuja etiquetas
	4	El usuario modifica etiquetas.
	5	El usuario borra etiquetas.
Postcondiciones	La imagen se muestra en el canvas del etiquetador.	
Excepciones	No hay cargada en el etiquetador ninguna imagen.	
Importancia	Alta	
Urgencia	Alta	
Comentarios		

Tabla B.2: Caso de uso 2: Etiquetado de fitolitos

Caso de uso 3: Generar datos de entrenamiento		
Descripción	El usuario puede guardar las imágenes y las etiquetas	
Requisitos	RF-3	
	RF-3.1	
	RF-3.2	
	RF-3.3	
	RF-3.4	
	RF-3.5	
Precondiciones	Ninguna	
Secuencia normal	Paso	Acción
	1	El usuario guarda las etiquetas e imágenes.
	2	El usuario ve dichas imágenes guardadas en la galería.
Postcondiciones	Las imágenes desaparecen del etiquetdor.	
Excepciones	No hay cargado en el etiquetador ninguna imagen.	
Importancia	Alta	
Urgencia	Alta	
Comentarios		

Tabla B.3: Caso de uso 3: Generar datos de entrenamiento

Caso de uso 2.1: Seleccionar imagen		
Descripción	El usuario selecciona una de las imágenes que haya subido	
Requisitos	RF-2	
	RF-2.1	
Precondiciones	Ninguna	
Secuencia normal	Paso	Acción
	1	El usuario selecciona una imagen subida.
Postcondiciones	La imagen se muestra en el canvas del etiquetador.	
Excepciones	No hay cargado en el etiquetador ninguna imagen.	
Importancia	Alta	
Urgencia	Alta	
Comentarios		

Tabla B.4: Caso de uso 2.1: Seleccionar imagen

Caso de uso 2.2: Seleccionar etiqueta		
Descripción	El usuario selecciona una etiqueta de la botonera	
Requisitos	RF-2	
	RF-2.1	
Precondiciones	Que haya imágenes subidas en el etiquetador	
Secuencia normal	Paso	Acción
	1	El usuario selecciona un tipo de etiqueta
Postcondiciones	Hay una etiqueta seleccionada y será el tipo con el que se etiquete hasta que se cambie.	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	
Comentarios		

Tabla B.5: Caso de uso 2.2: Seleccionar etiqueta

Caso de uso 2.3: Guardar etiquetas		
Descripción	Se generan los datos de las etiquetas	
Requisitos	RF-2	
	RF-2.1	
	RF-2.2	
	RF-2.3	
	RF-2.4	
	RF-2.5	
Precondiciones	Que haya imágenes subidas en el etiquetador	
Secuencia normal	Paso	Acción
	1	El usuario gestiona las etiquetas y la información se actualiza cuando deja de manipular cada etiqueta.
Postcondiciones	Ninguna	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	
Comentarios		

Tabla B.6: Caso de uso 2.3: Guardar etiquetas

Caso de uso 2.2.1: Dibujar etiquetas		
Descripción	Se dibujan las etiquetas sobre las imágenes	
Requisitos	RF-2.2	
Precondiciones	Que haya imágenes subidas en el etiquetador	
Secuencia normal	Paso	Acción
	1	El usuario dibuja las etiquetas sobre la imagen.
Postcondiciones	Ninguna	
Excepciones	Ninguna	
Importancia	Alta	
Urgencia	Alta	
Comentarios		

Tabla B.7: Caso de uso 2.2.1: Dibujar etiquetas

Caso de uso 2.2.2: Modificar etiquetas		
Descripción	Se modifica el tamaño o la posición de las etiquetas	
Requisitos	RF-2.3	
	RF-2.4	
Precondiciones	Que haya una etiqueta dibujada previamente	
Secuencia normal	Paso	Acción
	1	El usuario selecciona la etiquetas.
	2	El usuario mueve la etiqueta de sitio.
	3	El usuario redimensiona la etiqueta pinchando en el borde.
Postcondiciones	La etiqueta se ha movido o cambiado de tamaño	
Excepciones	Ninguna	
Importancia	Media	
Urgencia	Media	
Comentarios		

Tabla B.8: Caso de uso 2.2.2: Modificar etiquetas

Caso de uso 2.2.3: Borrar etiquetas		
Descripción	El usuario elimina la etiqueta seleccionada	
Requisitos	RF-2.2	
Precondiciones	Que haya una etiqueta dibujada previamente	
Secuencia normal	Paso	Acción
	1	El usuario selecciona la etiquetas.
	3	El usuario presiona el botón de suprimir o borrar.
Postcondiciones	La etiqueta se ha eliminado	
Excepciones	Ninguna	
Importancia	Media	
Urgencia	Media	
Comentarios		

Tabla B.9: Caso de uso 2.2.3: Borrar etiquetas

Apéndice C

Especificación de diseño

C.1. Introducción

En este apartado se van a presentar los aspectos más relevantes dentro del software generado.

C.2. Diseño de datos

En este proyecto vamos a trabajar con distintos tipos de datos:

- Ficheros en formato CSV, donde guardaremos las coordenadas y especificaciones de las etiquetas que se van a generar.
- Formatos de imágenes JPG, JPEG o PNG.

Las imágenes y las etiquetas se van a guardar dentro del servidor en una carpeta llamada
imágenes. El nombre de esta carpeta se debe a que, al principio, se pensaba hacer el almacenaje de imágenes y etiquetas en carpetas separadas, pero al final no se hizo porque para hacer la recogida de los ficheros *csv* y las propias imágenes, era más sencillo tenerlos ubicados en el mismo directorio.

Cada conjunto de imágenes que es etiquetado genera un solo archivo *csv* con el nombre de la imagen, el tipo de etiqueta, el tipo de fitolito y las coordenadas. En la imagen C.1 se muestra como se ven los datos si se abren con *LibreOffice Calc*.

	A	B	C	D	E	F	G
1	#filename	file_size	file_attributes	region_count	region_id	region_shape_attributes	region_attributes
2	bilobate2.PNG	19462	{}	3	0	{"name":"rect","x":32,"y":10,"width":38,"height":38}	{"tipo":"Bilobate"}
3	bilobate2.PNG	19462	{}	3	1	{"name":"rect","x":11,"y":64,"width":21,"height":14}	{"tipo":"Bilobate"}
4	bilobate2.PNG	19462	{}	3	2	{"name":"rect","x":56,"y":54,"width":29,"height":19}	{"tipo":"Rondel"}
5							

Figura C.1: Ejemplo de datos recogidos en un *csv*

Para dar una explicación más detallada sobre como están estructurados los datos del *csv* me serviré de la imagen C.1, previamente mencionada.

La primera columna contiene el nombre de la imagen sobre la que se ha etiquetado, la segunda columna contiene el tamaño de la imagen, la tercera columna tiene el número de etiquetas que hay dentro de esa imagen, la cuarta columna tiene el identificador de la etiqueta dentro de la imagen, por ejemplo, si una imagen tiene dos etiquetas, la primera etiqueta tendrá el identificador 0 y a segunda el identificador 1. La quinta columna tienes las coordenadas x e y de la esquina inferior izquierda de la etiqueta y el ancho y largo que va a tener esta. Por último, en la sexta columna, se tiene el tipo de fitolito que recoge.

C.3. Diseño procedimental

En el proyecto hay tres procedimientos significativos a destacar, los cuales se explicarán a continuación.

Procedimiento de lectura de imágenes y etiquetas para comprobación

En el anterior proyecto de reconocimiento de fitolitos [**?**], se generó un conjunto de etiquetas para un conjunto análogo de imágenes. Para comprobar si el clasificador hacía bien su trabajo, se hizo un script [1] para pintar las etiquetas sobre las imágenes.

El **primer procedimiento** corresponde a la lectura de datos que se hizo para su posterior representación. El pseudocódigo de dicho procedimiento se puede ver en el algoritmo 1.

[1] Se utilizó un *notebook* de *Jupyter* para esta tarea por simplicidad.

Algoritmo 1: Procedimiento para la comprobación del conjunto de etiquetas generado por el proyecto anterior.

```
1 para cada imagen en el directorio indicado hacer
2     si tiene un fichero JSON con el mismo nombre que la imagen
        entonces
3         Mostramos las etiquetas pintadas la imagen.
4     fin
5 fin
6 Comprobamos visualmente si alguna coincide con el resultado
    esperado.
```

Pasar a la siguiente imagen en el etiquetador

Dentro de la aplicación, cuando estamos trabajando en el etiquetador, podemos tener cargado un conjunto con varias imágenes. Cuando las vamos etiquetando y pasando a las siguientes, las etiquetas se van guardando en un archivo temporal. En el caso de que se seleccionase una imagen en la que se hubiese dibujado alguna etiqueta, el etiquetador deberá pintar esas etiquetas sobre la imagen. El procedimiento para esta función se puede ver en el algoritmo 2.

Algoritmo 2: Procedimiento para pasar a la siguiente imagen en el etiquetador

```
1 si Se cambia de imagen seleccionada entonces
2     Se carga la imagen en el canvas.
3     si Sobre esa imagen ya se han dibujado en esa sesión,
        previamente etiquetas. entonces
4         Se dibujan las etiquetas sobre la imagen
5         si Se modifica el conjunto de etiquetas entonces
6             Se actualizan los datos del conjunto de etiquetas.
7         fin
8     fin
9 fin
```

Procedimiento para la selección del tipo de fitolito

Este procedimiento se da en el etiquetador dentro de la aplicación. Si cuando se tienen cargadas las imágenes, antes de ponerte a dibujar, no

seleccionas ningún tipo de fitolito, el tipo por defecto será *bilobate*. El funcionamiento de este procedimiento se puede ver en el algoritmo 3

Algoritmo 3: Procedimiento para seleccionar el tipo de fitolito de una etiqueta

```
si Hay una imagen cargada en el canvas entonces
    si No se selecciona ningún tipo de fitolito entonces
        si Se dibuja una etiqueta entonces
            Tipo de la etiqueta = bilobate
        en otro caso
            Tipo de fitolito = Tipo seleccionado
        fin
    fin
fin
```

C.4. Diseño arquitectónico

En esta sección se van a explicar aquellos aspectos a destacar dentro del diseño de la arquitectura de la aplicación y la ordenación de los recursos usados.

Para este proyecto se ha seguido un diseño sencillo y accesible con dos finalidades:

- La primera, por facilitar la navegación entre carpetas durante el desarrollo del proyecto. De esta manera es más sencillo llevar a cabo cualquier tarea de búsqueda.
- Hacer más cómodas las dependencias entre archivos. De esta forma se evitan llamadas excesivamente largas. Además, a la hora de darle mantenimiento es bastante más sencillo.

Estructura de *Flask*

Hemos trabajado con *Flask* para realizar el despliegue de la aplicación en un servidor. *Flask* nos permite usar el código escrito en *Python* para realizar parte del desarrollo web.

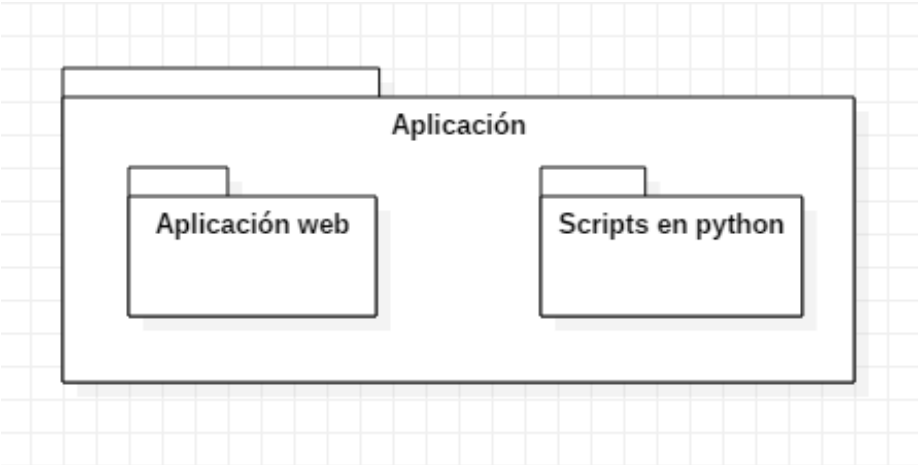

Figura C.2: Diagrama simplificado de la estructura funcional de la aplicación

Esta aplicación tiene dos partes diferentes (ver imagen C.2):

- Por un lado tenemos los componentes web de la aplicación (*HTML*,*JavaScipt, css...*).
- Y por otro lado tenemos los scripts de *Python* con *Flask* para el despliegue y para la comunicación con otros archivos escritos en *Python* que vayan a otorgar funcionale a la aplicación web.

En la parte de aplicación web podemos encontrar:

- La carpeta *images*: Contiene las imágenes que se guardan en local.
- La carpeta *statics*: contiene los ficheros en *JavaScript* y en *css*, los cuales serán usados desde los archivos *HTML*.
- La carpeta *templates*: la cual contiene los archivos *HTML* en los cuales se define la interfaz de la aplicación.
- *views.py*: *views.py* contiene la librería de *Flask*, con la cual se realizan las tareas de despliegue y de juntar los archivos de *Python* con el resto de la interfaz.

Los Scripts de *Python* son aquellos que corresponderían al clasificador.

El archivo *run.py* es el único archivo de *Python* que se mantiene alejado de la estructura y es debido a que es el script que se ejecutará cuando se

quiera usar la aplicación. Previamente, esta información hemos de dársela a los archivos de configuración del despliegue, los cuales se encuentran en la carpeta etc., y dentro del propio directorio [2].

C.5. Diseño de interfces

En este proyecto se ha desarrollado una página web, por lo que la interfaz es importante.

Para ello se ha hecho uso de *HTML* con *Bootstrap*, *JavaScript* y *css*.

La aplicación se ha dividido en cuatro secciones:

- **Inicio**: En él se muestra una breve introducción a las funcionalidades. Hay una barra de navegación en la parte superior con los diferentes botones que te llevan a las vistas correspondientes y una vista de la cuenta con la que se registrado para entrar y desde la cual se puede cerrar sesión. Dicha barra se mantendrá en todas las vistas y un pie de página con información del alumnos, de los tutores y el un enlace al repositorio, tal y como se muestra en la imagen C.3.

Figura C.3: página de inicio de la aplicación.

- **Etiquetador**: Desde el etiquetador se permite sibir imágenes, pintar etiquetas y guardarlas en el servidor. Los elementos como la barra de

[2]El fichero *boxfile.yml* se encuentra en el mismo directorio que *run.py*. Es imprtante porque contiene especificaciones del despliegue

navegación y el pie de página sigun igual que en la página de inicio. La apariencia del etiquetador será como la mostrada en C.4.

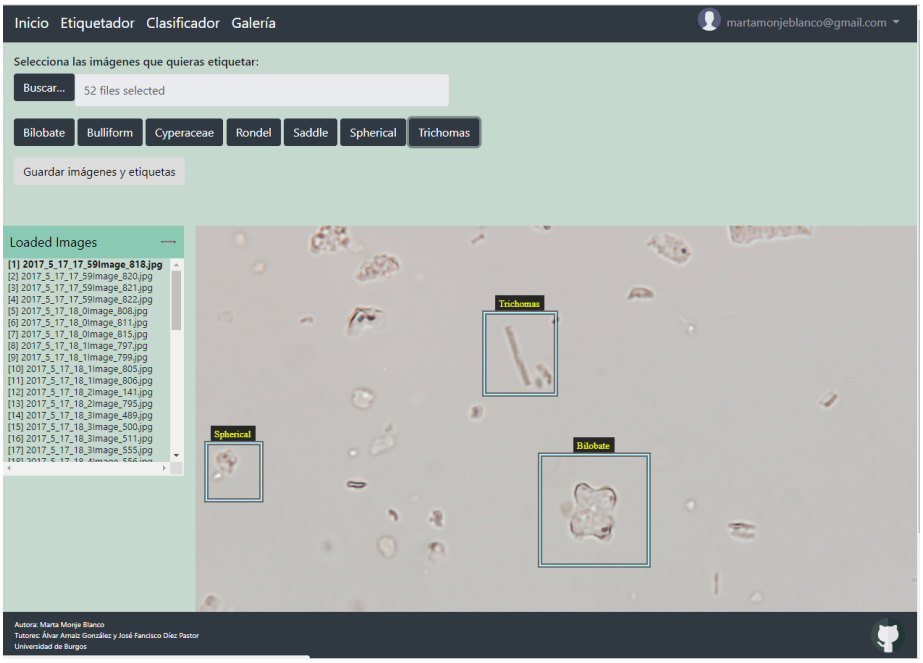

Figura C.4: Vista del etiquetador con imágenes cargadas y etiquetas dibujadas.

- **Galería**: En la galería, los elementos estáticos que se mantenían en el resto de vistas, como el pie de página, seguirán siéndolo. Esta vista muestra una galería con las imágenes que se subieron previamente al servidor. Su apariencia es como la mostrada en la imagen E.11.

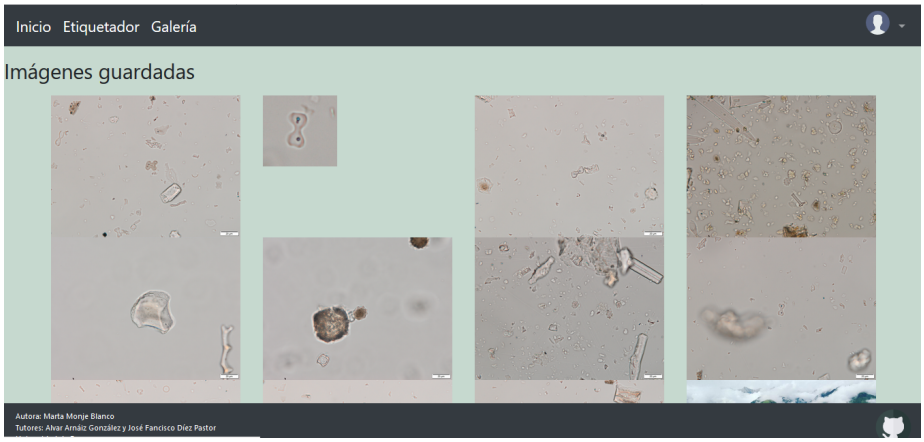

Figura C.5: Muestra de como se ven las imágenes dela galería.

Apéndice D

Documentación técnica de programación

D.1. Introducción

En este apéndice se van a explicar los requisitos y directorios del proyecto y la documentación para que otra persona retome el desarrollo.

D.2. Estructura de directorios

Las estructura de carpetas del proyecto se encuentra organizada en forma de árbol. En este apartado se explicarán brevemente.

- **Directorio raíz**: En este directorio tenemos el *.gitignore*, el *README* y la licencia del proyecto.
    - **code**: contiene los scripts con la mayor parte de la lógica de la aplicación.
        - **notebook**: ontiene el *notebook* con las comprobaciones de las etiquetas del proyecto anterior.
        - **pytolithclassifier**: en este directorio podemos encontrar el fichero *boxfile.yml*, uno de los más importantes debido a que es necesario para el despliegue de la aplicación. También se encuentra el *Procfile*, *requirements.txt* con las librerías y las versiones que hay que instlar. El fichero *run.py*, el cual hay que ejecutar para poner en marcha la aplicación y el *runtime.txt*.

- ◇ **.idea**: Archivos referentes al delpliegue.
- ◇ **app**: Contiene los ficheros en *HTML, JavaScript y CSS* además de *views.py*, que va a tener la librería de *Flask* en uso y *init.py* que se encarga de pasar los datos de ejecución al *run.py*.

  images: contiene las imágenes que se han guardado en local.

  static: contiene dos carpetas llamadas *js* y *css* y contienen los archivos en *JavaScript* y *css* respectivamente.

  templates: contiene los ficheros *HTML*
- ◇ **etc**: contiene los ficheros *gunicorn.py* y *nginx.conf*, los cuales se usan en el despliegue.
- ◇ **recursos del clasificador**: Contiene aquellos recursos que se pudieron conseguir en el avance con el clasificador, como los *csv* de entrenamiento, los scripts de conversión, el mapa de etiquetas y diversos archivos de configuración.

- **Docs**: Contiene la documentación del proyecto, anexos y memoria.
  - **img**: contiene las imágenes empleadas en la documentación.
  - **tex**: contiene los diferentes apartados de anexos y memoria.
- **Lecturas y enlaces**: contiene un fichero escrito en *marckdown*[1]

D.3. Manual del programador

En este apartado se van a introducir aspectos importantes a tener en cuenta dentro de la continuación con el desarrollo.

Librerías usadas

Para facilitar la labor de instalar librerías se ha facilitado dentro de la carpeta de *pytolithclassifier* un documento *txt* con todas las librerías con las versiones especificadas. Si bien es cierto que es mejor trabajar con las librerías actualizadas, sería recomendable probar primero el proyecto con las versiones recomendadas y después ya pensar en actualizarlas.

[1]Está hecho en *marckdown* porque se editaba desdeel repositorio de *GitHub*. Para saber más de *marckdown*: https://es.wikipedia.org/wiki/Markdown

Flask

Esta librería es la más importante que se ha usado, ya que es la piedra angular que encaja la lógica escrita en *Python* con la interfaz escrita en *HTML, JavaScript y css*.

CUDA y *Tensorflow*

Antes de intentar hacer nada del clasificador, es necesario aclarar que se van a necesitar tanto CUDA, para el procesamiento de imágenes, como *Tensorflow* para entrenar al modelo.

El problema está en que las versiones más nuevas de ambas son incompatibles[2].

Clasificador

Recomiendo seguir este tutorial [**?**] por varias razones.

La primera razón es que, si no tienes ningún problema, la cantidad de trabajo requerida es poca, con lo cual, podrás aprender a crear un clasificador de imágenes en un tiempo reducido.

Otra razón es que viene muy bien explicado, el instructor se preocupa porque cada explicación sea minuciosa, no hace algo sin contar por qué.

Otra razón es que te deja disponible todos los recursos en un su repositorio y puedes seguir el tutorial por vídeo si así lo deseas, no obstante, recomiendo seguir el tutorial a través del repositorio, ya que se explican más cosas.

Muchos de los recursos necesarios para este tutorial ya están hechos y guardados en la carpeta de *recursos del clasificador*, lo cual puede o bien servir de ejemplo o ser usados.

D.4. Compilación, instalación y ejecución del proyecto

Si se quiere ejecutar en local, una vez se tenga bien adecuado el entorno de *Python* hay que ejecutar el fichero *run.py*, el cual se encuentra dentro de la carpeta *code/pytolithclassifier*.

[2]Hay muchos foros que tratan esta temática, pero en este hilo está muy bien explicado: https://stackoverflow.com/questions/50622525/which-tensorflow-and-cuda-version-combinations-are-compatible

No hay nada que instalar más allá de tener in IDE para *Python* y un entorno de *Python* adecuado, preferiblemente *Anaconda*.

El enlace de la página de la aplicación se encuentra en el repositorio, solo hay que pincharle para acceder.

D.5. Pruebas del sistema

Apenas se han realizado pruebas a lo largo del proyecto debido a la falta de tiempo. Realmente cuando se está desarrollando un producto de software es decuado realizar pruebas desde los primeros pasos del desarrollo.

Sí que se ha puesto a prueba el etiquetador con diferentes imágenes de distintos tamaños para comprobar que escalaba bien.

Se hicieron pruebas en la subida de imágenes, para comprobar que solo se subían determinados formatos.

Sería interesante plantear las pruebas en la siguiente parte del proyecto y utilizar alguna herramienta que lo automatice como *Selenium*[3].

[3]Para la automatización de teses en páginas web: https://www.seleniumhq.org/

Apéndice E

Documentación de usuario

E.1. Introducción

En este anexo se explicará de forma detalada como ejecutar la aplicación desde el punto de vista de un usuario medio.

E.2. Requisitos de usuarios

El único requisito que hay que tener en cuenta es que se necesita conexión a internet para ejecutar la aplicación.

E.3. Instalación

No precisa de instalación alguna[1], es online.

Se puede acceder a la aplicación desde aquí: http://martamonjeblanco.pythonanywhere.com/

E.4. Manual del usuario

En este apartado se van a explicar los pasos a seguir para usar la aplicación.

[1]Es una de las razones por las que se decidió hacer online, porque no se necesita tener nada instalado y es más fácil para el usuario

Identificación de usuario

Antes de entrar a la aplicación, se e solicitará al usuario acceder a través de una cuenta de *Google* tal y como se muestra en la imagen E.1.

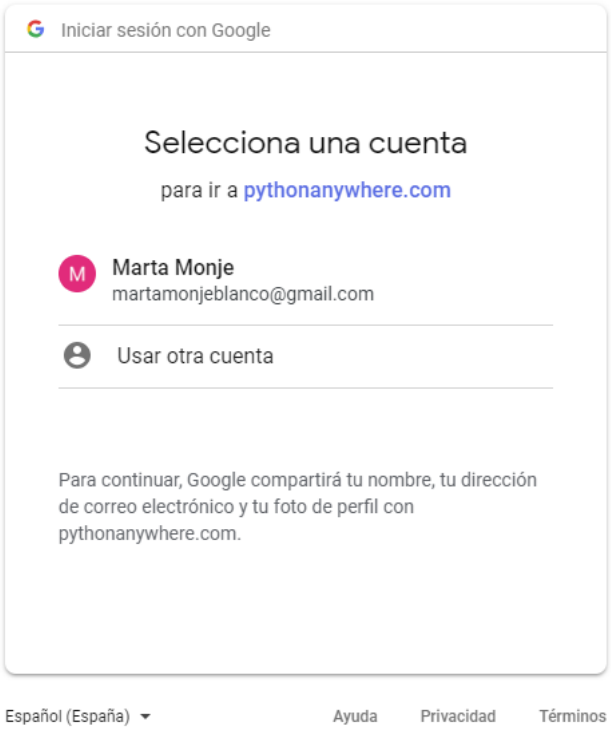

Figura E.1: Identificación de usuarios po medio de *Google*

Navegación

Para cambiar de una vista a otra solo hay que seleccionar a que vista se quiere acceder.

La barra de navegación aparece en la parte superior de la pantalla y se ve como en la imagen E.2

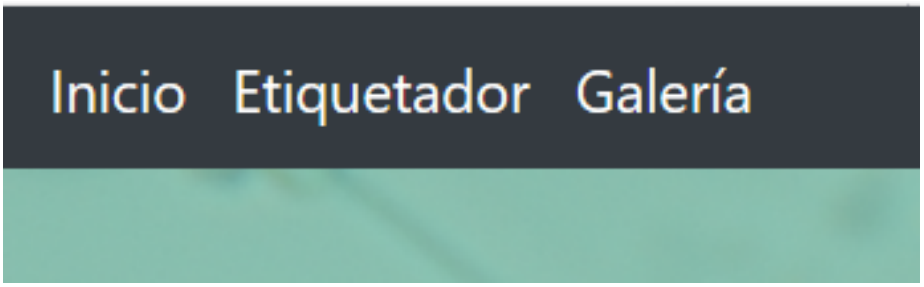

Figura E.2: Barra de navegación de la aplicacón

Inicio

La pantalla de inicio se verá así C.3.

En ella el usuario podrá cerrar sesión como se muestra en la imagen E.3

Viene una pequeña descripción de la aplicación.

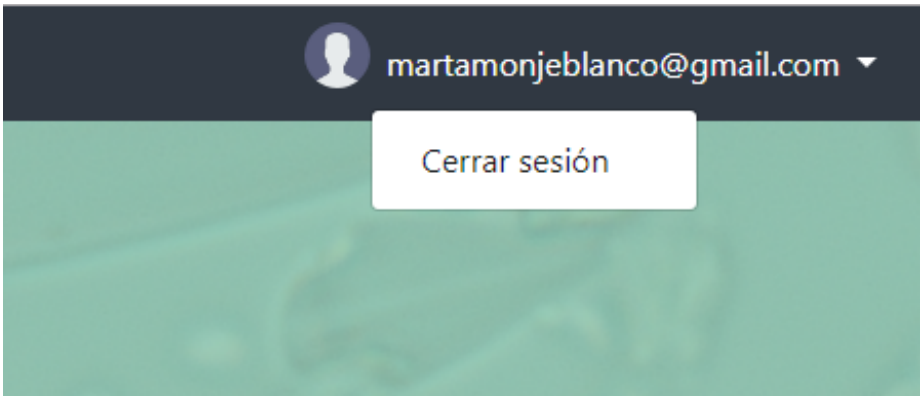

Figura E.3: Opción de cerrar sesión en el caso de que se desee salir o cambiar de cuenta.

Etiquetador

Cuando se accede al etiquetador la vista que se tiene de él es como la de la imagen E.6.

Buscar imágenes

Hay que pinchar sobre el botón de *Buscar...* ubicado en la parte superior a la izquierda (ver imagen E.4).

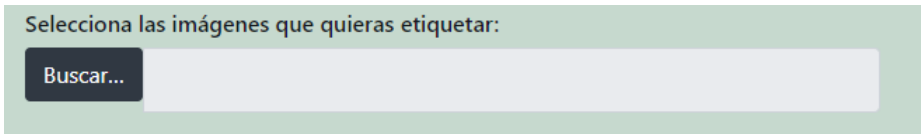

Figura E.4: Botón de buscar imágenes en el equipo.

Seleccionar imágenes

Cuando se pincha en el botón *Buscar...*, se abre una ventana mostrando los archivos de tipo imagen que tenemos en nuestro dispositivo tal y como se muestra en la imagen E.5.

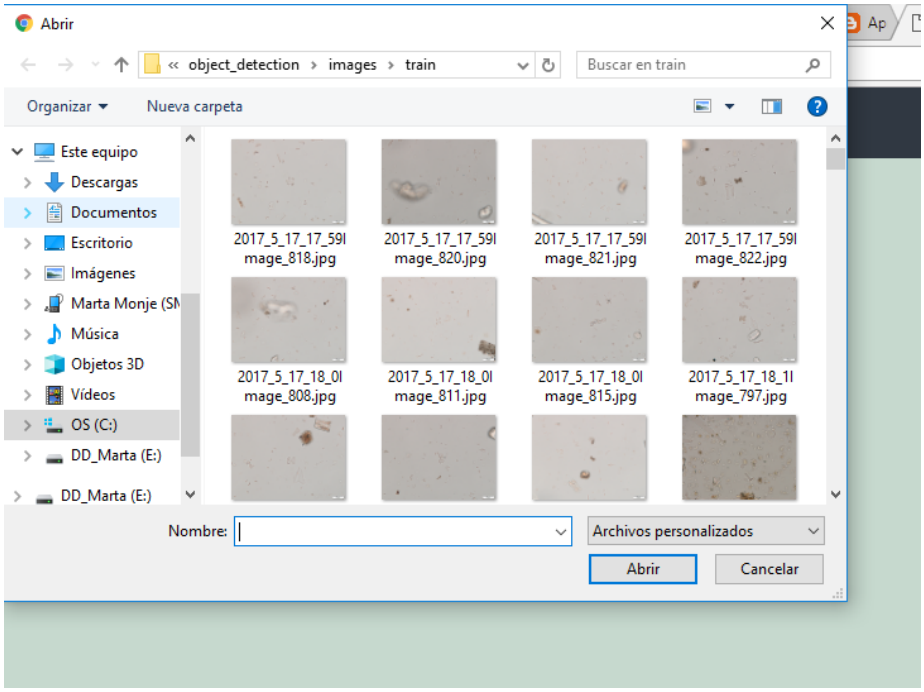

Figura E.5: Seleccionar la imagen o el conjunto de imágenes que se desee.

Solo se pueden seleccionar archivos de tipo imagen, para evitar posibles errores.

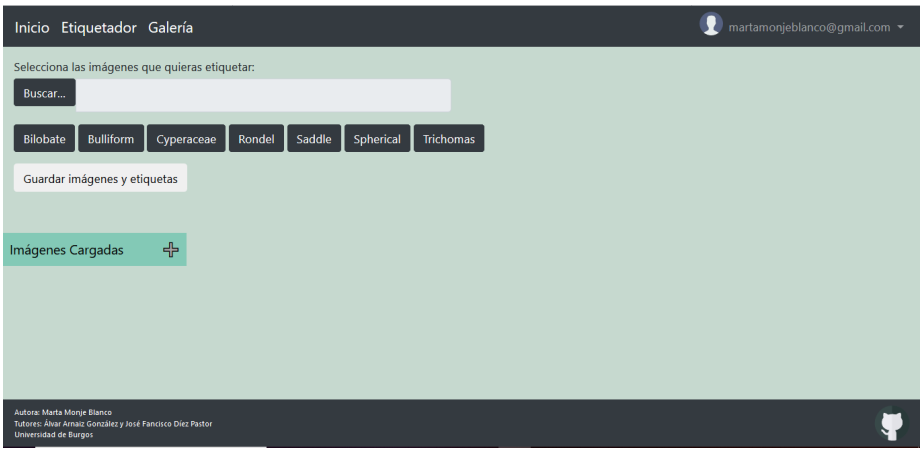

Figura E.6: Etiquetador sin imágenes cargadas

Una vez estén elegidas as imágenes, hay que darle al botón de *Abrir* y automáticamente cargará las imágenes en la aplicación y cerrara la ventana de selección de imágenes.

Imágenes cargadas

Las imágenes cargadas se van a mostrar en el lateral izquierdo como se ve en la imagen E.7

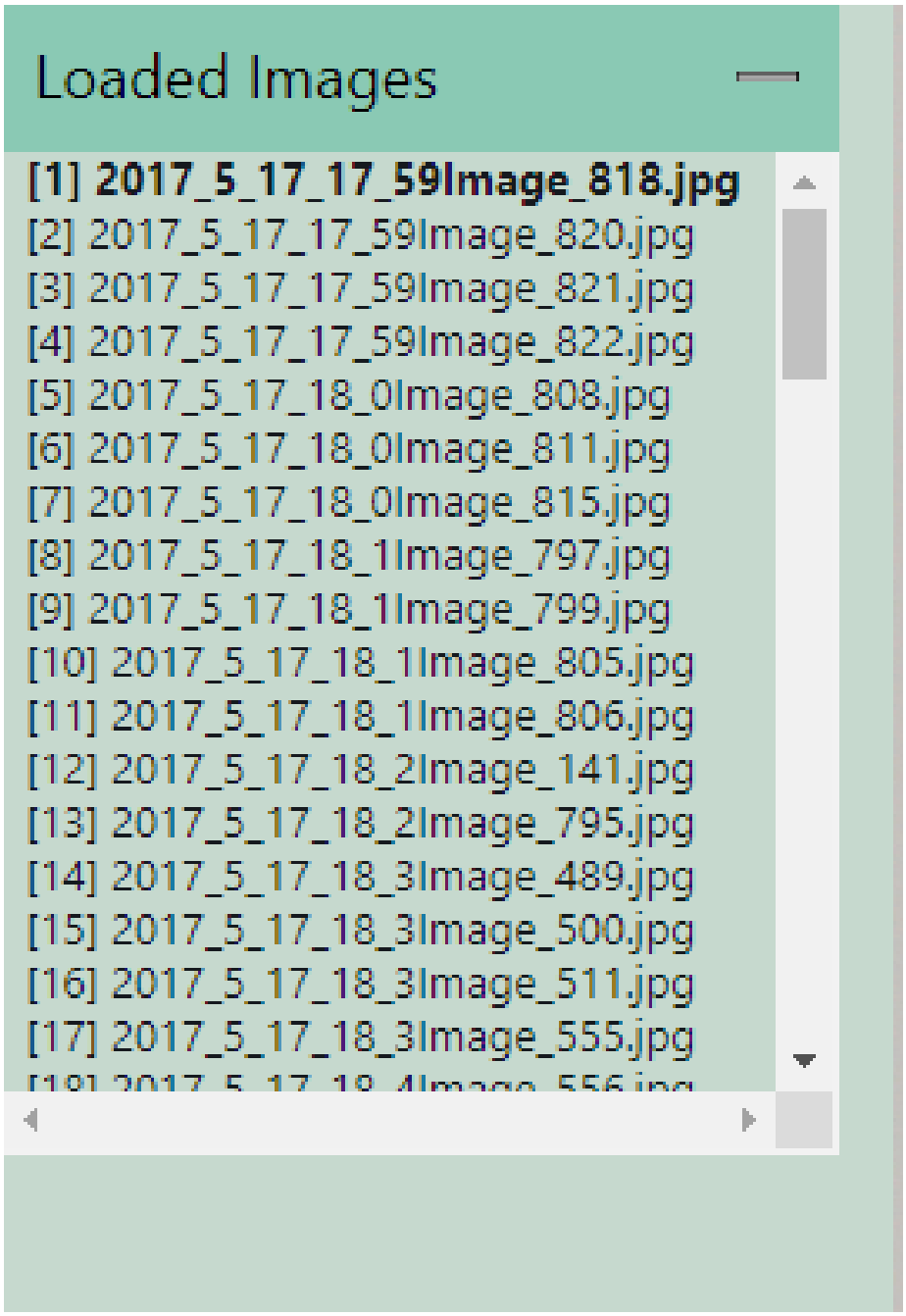

Figura E.7: Imágenes que se han cargado en la aplicación pero que aún no se han subido

Botones de etiquetas

Antes de ponerse a etiquetar hay que seleccionar el tipo de fitolito en la botonera (ver imagen E.8) de la parte superior.

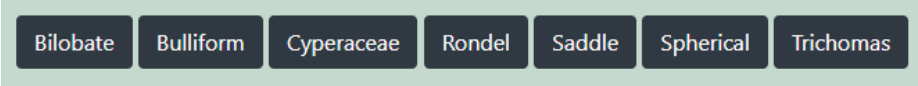

Figura E.8: Botonera para seleccionar el tipo de fitolito que se va a etiquetar.

Si no seleccionase ninguno se establecería el *bilobate* como el fitolito por defecto.

Dibujar etiquetas

Para dibujar etiquetas tenemos que tener una imagen cargada en el canvas. Se pincha y se arrastra para generar una etiqueta restangular sobre la imagen y se le asignará el tipo de fitolito que se haya elegido.

Modificar etiquetas

Para modificar una etiqueta podemos cambiarla de sitio dentro de la imagen o redimensionarla.

Para cambiarla de sitio hay que seleccionar la imagen, como se ve en la imagen E.9 y pinchar sin soltar hasta que se esté conforme con la nueva ubicación

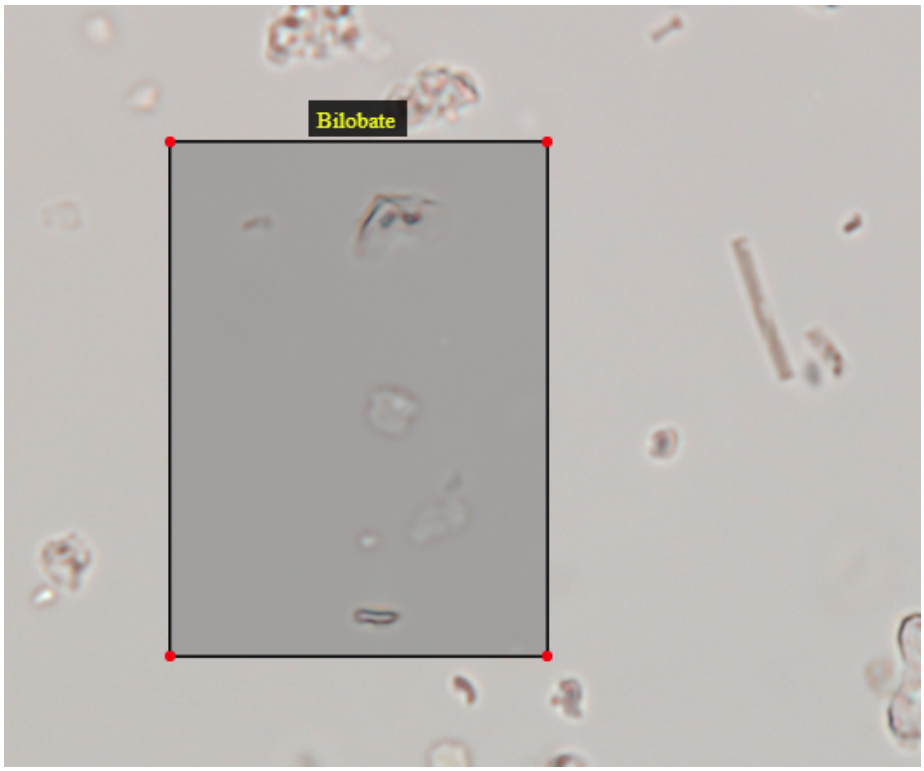

Figura E.9: Etiqueta seleccionada

Si se quisiera redimensionar, se tendría que seleccionar la etiqueta y pinchar en el borde o esquina que se quiera cambiar.

Copiar

Si se desea copia una etiqueta, con el tipo de fitolito incluido, solo hay que seleccionar una etiqueta y hacer *ctrl + c ctrl + v* ya ya se tiene una copia.

Eliminar etiquetas

Para elimirar las etiquetas que no se quieran en la imagen o porque se ha cometido un error, Hay que seleccionar la etiqueta y darle al botón de *suprimir* o al de *borrar*.

Guardar etiquetas

Una vez se ha acabado de etiquetar las imágenes podemos guardar las etiquetas y las imágenes para su posterior uso.

Hay que pulsar el botón de *Guardar imágenes y etiquetas* que se ve como se muestra en la imagen E.10.

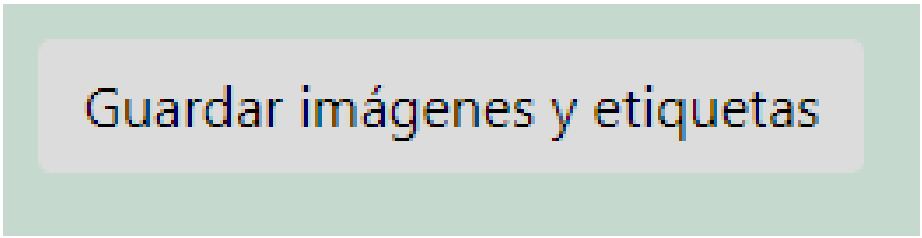

Figura E.10: Botón de guardado de imágenes y etiquetas.

Se guardaran las etiquetas en descargas.

Galería

La opción de la galería es meramente informativa. En ella se muestran las imágenes que se muestran las imágenes que se han subido al servidor como se ve en la imagen E.11

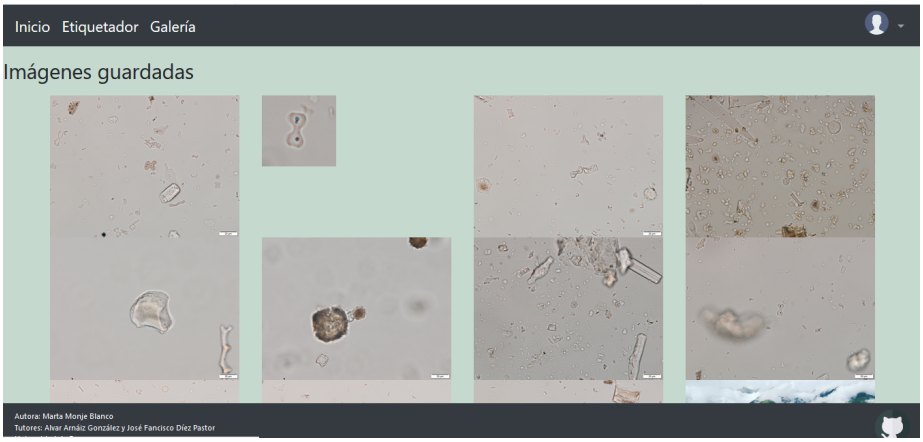

Figura E.11: Vista de la galería con las imágenes del servidor

title

Bibliografía