



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**UBUSETAS 1.0
Documentación Técnica**



Presentado por Adrián Antón García
en Universidad de Burgos — 10 de enero
de 2018

Tutor:

Dr. José F. Díez Pastor
Dr. Raúl Marticorena Sánchez

Índice general

Índice general	I
Índice de figuras	III
Índice de tablas	v
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	1
A.3. Estudio de viabilidad	24
Apéndice B Especificación de Requisitos	27
B.1. Introducción	27
B.2. Objetivos generales	27
B.3. Catalogo de requisitos	28
B.4. Especificación de requisitos	30
Apéndice C Especificación de diseño	43
C.1. Introducción	43
C.2. Diseño de datos	43
C.3. Diseño procedimental	56
C.4. Diseño arquitectónico	58
C.5. Diseño de la interfaz	62
Apéndice D Documentación técnica de programación	77
D.1. Introducción	77
D.2. Estructura de directorios	77

D.3. Manual del programador	78
D.4. Compilación, instalación y ejecución del proyecto	83
D.5. Pruebas del sistema	86
Apéndice E Documentación de usuario	89
E.1. Introducción	89
E.2. Requisitos de usuarios	89
E.3. Instalación	89
E.4. Manual del usuario	89
Bibliografía	91

Índice de figuras

A.1.	Gráfico Burndown del sprint 1	3
A.2.	Gráfico Burndown del sprint 2	4
A.3.	Gráfico Burndown del sprint 3	5
A.4.	Gráfico Burndown del sprint 4	6
A.5.	Gráfico Burndown del sprint 5	7
A.6.	Issues del sprint 5	8
A.7.	Gráfico Burndown del sprint 6	9
A.8.	Issues del sprint 6	9
A.9.	Gráfico Burndown del sprint 7	11
A.10.	Issues del sprint 7	11
A.11.	Gráfico Burndown del sprint 8	12
A.12.	Issues del sprint 8	13
A.13.	Gráfico Burndown del sprint 9	14
A.14.	Issues del sprint 9	14
A.15.	Gráfico Burndown del sprint 10	15
A.16.	Issues del sprint 10	16
A.17.	Gráfico Burndown del sprint 11	17
A.18.	Issues del sprint 11	17
A.19.	Gráfico Burndown del sprint 12	18
A.20.	Issues del sprint 12	19
A.21.	Gráfico Burndown del sprint 13	20
A.22.	Issues del sprint 13	21
A.23.	Gráfico Burndown del sprint 14	22
A.24.	Issues del sprint 14	23
B.1.	Diagrama de casos de uso del administrador	31
B.2.	Diagrama de casos de uso del usuario	32

C.1. Tablas de la base de datos SQLite.	44
C.2. Diagrama de clases del proyecto de Eclipse.	47
C.3. Diagrama de clases del paquete basedatos.	51
C.4. Diagrama de clases del paquete clasificador.	51
C.5. Diagrama de clases del paquete clavedicotomica.	52
C.6. Diagrama de clases del paquete elegirclaves.	52
C.7. Diagrama de clases del paquete informacion.	53
C.8. Diagrama de clases del paquete lanzador.	53
C.9. Diagrama de clases del paquete resultados.	54
C.10. Diagrama de clases del paquete tarjetasclaves.	54
C.11. Diagrama de clases del paquete tarjetassetas.	55
C.12. Diagrama de paquetes del proyecto de Eclipse.	60
C.13. Diagrama de paquetes del proyecto Android.	61
C.14. Prototipo actividad Lanzadora.	62
C.15. Prototipo actividad Recoger Foto.	63
C.16. Prototipo actividad Mostrar Claves.	64
C.17. Prototipo actividad Mostrar Setas.	65
C.18. Prototipo actividad Mostrar Resultados.	66
C.19. Prototipo actividad Clave Dicotómica.	67
C.20. Prototipo Mostrar Información.	68
C.21. Prototipo Elegir Claves.	69
C.22. Prototipo Mostrar Comparativa.	70
C.23. Diagrama de flujo.	71
C.24. Vertical sin menú.	72
C.25. Vertical con menú.	73
C.26. Horizontal sin menú.	74
C.27. Horizontal con menú.	74
D.1. Ruta de la base de datos.	79
D.2. Listado de especies de setas.	80
D.3. Estructura Image Retraining.	81
D.4. Estructura Image Retraining.	82
D.5. Activación Tensorflow.	82
D.6. Entrenamiento Mobilenet.	83
D.7. Importación Eclipse.	84
D.8. Importación Eclipse.	84
D.9. PruebasUnitarias.	87
D.10. <i>Coverage</i> pruebas unitarias.	87
D.11. Ejemplo prueba integración.	88

Índice de tablas

A.1. Costes de personal	25
A.2. Costes totales	25
A.3. Licencias	26
B.1. Caso de uso 1: Adquirir información especies.	32
B.2. Caso de uso 2: Entrenar el clasificador.	33
B.3. Caso de uso 3: Adquirir claves dicotómicas.	34
B.4. Caso de uso 4: Generar Base de datos.	35
B.5. Caso de uso 5.1: Clasificar imagen.	36
B.6. Caso de uso 5.2: Guardar imagen.	37
B.7. Caso de uso 5.3: Ver resultados del clasificador.	38
B.8. Caso de uso 5.4: Contestar preguntas clave dicotómica.	39
B.9. Caso de uso 5.5: Ver información de una especie de seta.	40
B.10. Caso de uso 5.6: Introducir imagen.	41
B.11. Caso de uso 5.7: Acceder a la ayuda.	42

Apéndice A

Plan de Proyecto Software

A.1. Introducción

Sección en la que se va a desarrollar la planificación seguida en el proyecto, así como su viabilidad legal y económica.

Para llevar a cabo el seguimiento y planificación del proyecto se va a seguir la metodología *Scrum* mediante Github, realizando un desarrollo incremental, dividido en sprints de una semana de duración cada uno. Durante cada sprint se irán creando y realizando las diferentes tareas correspondientes a los objetivos fijados en cada sprint. Al final de cada sprint se van a realizar las reuniones con los tutores para fijar las nuevas tareas de los nuevos sprints.

Para organizar las diferentes tareas se usará el tablero Kanban de Zenhub donde se mostrarán los diferentes estados de desarrollo en los que se encuentran las tareas.

El repositorio con el proyecto y las issues realizadas se puede encontrar en el siguiente enlace: <https://github.com/AdrianAntonGarcia/-TFG-UBUSetas>

A.2. Planificación temporal

En esta sección se va a explicar la planificación del proyecto a través de los diferentes sprints realizados, explicando las fechas en las que se desarrollaron y que tareas se realizaron en cada uno. Los primeros cuatro sprint no se muestran de forma correcta debido a que no se cerraron las tareas de forma adecuada en esos sprint, este problema se arregló a partir del quinto sprint.

Cada sprint estará acompañado de su grafo Burndown y un listado de las tareas realizadas. El proyecto se inició el 11 de septiembre de 2017 y está planeado entregarse en el 14 de enero de 2018, planificando así su desarrollo en cuatro meses.

Sprint 1

Este sprint se desarrolló entre los días 11 y 17 de septiembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Estudiar las propuestas de los tutores para elegir como desarrollar el clasificador.
- Empezar a estudiar las herramientas disponibles para entrenar los modelos.
- Elegir una propuesta para crear el clasificador.

La primera semana del proyecto se dedicó a estudiar las diferentes técnicas propuestas para implementar el clasificador de imágenes, así como pensar las diferentes ventajas e inconvenientes de estas implementaciones y comprobar si eran viables con respecto al hardware y medios disponibles. Estas propuestas eran:

- Crear un clasificador que se ejecute en un servidor y devuelva los resultados a la aplicación móvil. El modelo se puede reentrenar o entrenar desde cero.
- Reentrenar un modelo mediante la herramienta Tensorflow para poder ejecutar este modelo en el propio móvil.



Figura A.1: Gráfico Burndown del sprint 1.

Sprint 2

Este sprint se desarrolló entre los días 18 y 24 de septiembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Estudiar las librerías de Tensorflow para el entrenamiento de los clasificadores.
- Estudiar los modelos Mobilenet e Inception.
- Estudiar como reentrenar los modelos mencionados.
- Estudiar como implementar los modelos en Android.

Esta semana relacionada con el segundo sprint se dedicó principalmente al estudio de las diferentes técnicas disponibles para entrenar modelos que pudieran funcionar en una arquitectura móvil. Para ello se realizó el estudio de las librerías de Tensorflow y se siguieron los ejemplos para reentrenamiento disponibles en el repositorio de Github público de Tensorflow.

Además se estudiaron las capacidades Hardware de las que disponíamos para entrenar estos modelos, ya que son procesos que requieren bastante tiempo de ejecución y había que comprobar si podíamos reentrenar con los equipos disponibles.



Figura A.2: Gráfico Burndown del sprint 2.

Sprint 3

Este sprint se desarrolló entre el día 25 de septiembre y 1 de octubre de 2017. Se realizaron las siguientes tareas y objetivos:

- Estudiar el entorno de programación Android Studio.
- Aprender a programar en Android.
- Implementar los modelos de clasificación en una aplicación Android.

El tercer sprint se dedicó principalmente a familiarizarme a desarrollar en Android. El objetivo era el de tener una base para poder ir implementando los modelos Mobilenet e Inception en una aplicación de prueba Android.

Tras esta semana se consiguió una primera demo que ejecutaba un modelo Mobilenet en el propio dispositivo móvil, aunque no funcionaba correctamente y tenía errores que se corrigieron según se fue profundizando en los diferentes aspectos a estudiar.



Figura A.3: Gráfico Burndown del sprint 3.

Sprint 4

Este sprint se desarrolló entre los días 2 y 8 de octubre de 2017. Se realizaron las siguientes tareas y objetivos:

- Estudiar en qué consistía y cómo realizar consultas a una web semántica.
- Estudiar el marco de definición de recursos *RDF*.
- Estudiar el lenguaje de consultas *SPARQL*, para realizar las consultas a una web semántica.
- Estudiar el funcionamiento de la *DBpedia* como Web semántica.
- Estudio de la herramienta *Apache Jena* para realizar estas consultas a través de un programa Java.

Esta semana tenía como objetivo el entender en qué consistía una web semántica y cómo se podía implementar un programa Java que realizara consultas a una web semántica. El objetivo era el de estudiar como crear un programa Java que recopilara, de manera automática, información de las diferentes especies de setas.

Esta tarea tenía la complicación de que necesitábamos una web semántica que contuviera información de todas las setas, para poder facilitar la tarea de automatizar la extracción de información, y que esta se mostrará de manera uniforme. Tras estudiar las opciones disponibles se optó por la DBpedia ya que era la única que cumplía con las especificaciones necesarias y que contenía información suficiente de todas las especies.



Figura A.4: Gráfico Burndown del sprint 4.

Estos cuatro primeros sprints se dedicaron a estudiar las diferentes herramientas necesarias para implementar las propuestas del proyecto. La mayoría de estas herramientas eran desconocidas para mí por lo que me llevó un tiempo familiarizarme con ellas y empezar a implementar los diferentes programas.

En estos primeros sprints se consiguió crear en Android una aplicación que ejecutaba los modelos Mobilenet e Inception y en Java una aplicación que realizaba parte de las consultas necesarias a la DBpedia.

Sprint 5

Este sprint se desarrolló entre los días 11 y 18 de octubre de 2017. Se realizaron las siguientes tareas y objetivos:

- Estudiar como implementar una base de datos SQL que funcione tanto en Android como en Windows.

- Implementar la base de datos en Windows para el programa de consultas a la web semántica.
- Implementar base de datos en la aplicación Android.
- Elegir una base de datos apropiada para los requisitos de la aplicación.

Para implementar la base de datos, se empezó estudiando e implementando una base de datos Microsoft JDBC en Windows con la intención de exportarla posteriormente a Android. Aunque se consiguió realizar esta parte, se decidió cambiar a una base de datos SQLite, ya que era mucho más ligera que la de Microsoft y más sencilla de usar para las necesidades básicas que tiene nuestra aplicación.

En esta semana se consiguió crear la base de datos y los métodos de acceso necesarios tanto en la aplicación de Eclipse como en la Android.



Figura A.5: Gráfico Burndown del sprint 5.

Completed Issues	Story points
S5-1.Realizar un estudio previo de SQLite tanto en Windows como Android Estudio -TFG-UBUSetas #14 III New Issues ↑ Sprint-5	(2)
S5-2.Realizar la instalación de SQLite en Windows Instalación -TFG-UBUSetas #15 III New Issues ↑ Sprint-5	(2)
S5-3.Crear una base de datos SQLite para almacenar los contenidos de la Web Sem... Desarrollo -TFG-UBUSetas #16 III New Issues ↑ Sprint-5	(3)
S5-4.Conectar la base de datos SQLite a los métodos de la Web Semántica ya creados. Desarrollo -TFG-UBUSetas #17 III New Issues ↑ Sprint-5	(1)
S5-5.Crear los métodos necesarios para modificar la base de datos SQLite en eclips... Desarrollo Test -TFG-UBUSetas #18 III New Issues ↑ Sprint-5	(6)
S5-6.Realizar la instalación de SQLite en Android Instalación -TFG-UBUSetas #19 III New Issues ↑ Sprint-5	(2)
S5-7.Exportar la base de datos de Windows para ser usada en Android Desarrollo -TFG-UBUSetas #20 III New Issues ↑ Sprint-5	(1)
S5-8.Creación de los métodos necesarios para acceder a la base de datos en android. Desarrollo -TFG-UBUSetas #21 III New Issues ↑ Sprint-5	(5)
S5-9.Importar la base de datos de Windows a Android Desarrollo Test -TFG-UBUSetas #22 III New Issues ↑ Sprint-5	(3)
S5-10.Documentación de todos los métodos creados en este sprint. Documentación -TFG-UBUSetas #23 III New Issues ↑ Sprint-5	(2)

Figura A.6: Issues del sprint 5.

A partir de este sprint se empezó a utilizar de forma más correcta la Herramienta Zenhub y a mostrar de forma correcta las tareas desarrolladas en cada sprint.

Sprint 6

Este sprint se desarrolló entre los días 18 y 25 de octubre de 2017. Se realizaron las siguientes tareas y objetivos:

- Crear la aplicación Java de acceso a la DBpedia a partir de las pruebas creadas.
- Crear la primera versión de la aplicación Android que muestre los resultados del clasificador.
- Crear un método en Java que traduzca textos de manera automática.
- Preparar la aplicación Android para mostrar los datos recibidos de la aplicación Java.

En esta semana se trabajó paralelamente tanto en la aplicación Android como en las consultas a la DBpedia. Se creó una versión Android que

incorporaba las primeras actividades definidas en el prototipado. Se integró la información recogida por la aplicación Android en la base de datos de la aplicación Android.

Por último se creó un método Java que traducía realizando llamadas al traductor de Google, con el fin de entregar una aplicación internacionalizada tanto al español como el inglés.



Figura A.7: Gráfico Burndown del sprint 6.

Completed Issues	Story points
S6-1.Creación de la actividad para comparar imágenes de setas. Desarrollo Mejora -TFG-UBUSetas #24 III New Issues ↑Sprint-6	(3)
S6-2.Creación de la actividad para mostrar información. Desarrollo Mejora -TFG-UBUSetas #25 III New Issues ↑Sprint-6	(2)
S6-3.Modificación de la actividad mostrar resultados Desarrollo Mejora -TFG-UBUSetas #26 III New Issues ↑Sprint-6	(2)
S6-4.Modificación de la creación de la base de datos Desarrollo Mejora -TFG-UBUSetas #27 III New Issues ↑Sprint-6	(5)
S6-5.Creación de las nuevas consultas a la DBpedia Desarrollo -TFG-UBUSetas #28 III New Issues ↑Sprint-6	(8)
S6-6.Creación de un método traductor Desarrollo Estudio -TFG-UBUSetas #29 III New Issues ↑Sprint-6	(8)
S6-7.Incorporación de link Desarrollo -TFG-UBUSetas #30 III New Issues ↑Sprint-6	(2)

Figura A.8: Issues del sprint 6.

Sprint 7

Este sprint se desarrolló entre el día 25 de octubre y 1 de noviembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Estudiar como realizar Web Scraping en una página Web.
- Estudiar la herramienta *Jaunt* para realizar Web Scraping en Java.
- Crear los métodos necesarios en el proyecto de Eclipse para extraer las claves dicotómicas de la página web <http://www.avelinosetas.info/claves.php>.
- Crear las actividades necesarias en la aplicación Android para mostrar las claves dicotómicas.
- Crear los métodos necesarios tanto en el proyecto Android y Eclipse para serializar las claves dicotómicas y transferirlas desde el proyecto de Eclipse a la aplicación Android.

En este sprint se encontró una página web con una clave dicotómica que contenía una gran cantidad de géneros de los clasificados por el clasificador, aunque no todos y mostraba claves de géneros para clasificar especies concretas de setas. Además la estructura html se repetía en todas las claves, lo que facilitó la aplicación de las técnicas de Web Scraping.

Se decidió serializar las claves dicotómicas en estructuras de datos Java para exportar de manera sencilla las claves a la aplicación Android, sin necesidad de tener que usar la base de datos SQLite.



Figura A.9: Gráfico Burndown del sprint 7.

Completed Issues	Story points
S7-1.Creación de método que extraiga las claves dicotómicas Desarrollo Eclipse Primera Versión -TFG-UBUSetas #31 III New Issues ↑Sprint-7	(8)
S7-2.Creación de un método que serialice las claves en un archivo Desarrollo Eclipse Primera Versión -TFG-UBUSetas #32 III New Issues ↑Sprint-7	(2)
S7-3.Creación de una actividad en Android que visualice una clave completa Android Desarrollo Primera Versión -TFG-UBUSetas #33 III New Issues ↑Sprint-7	(3)
S7-5.Modificación del repositorio de imágenes de setas Desarrollo Instalación Mejora -TFG-UBUSetas #35 III New Issues ↑Sprint-7	(8)
S7-6.Creación del nuevo clasificador con el nuevo repositorio de imágenes Desarrollo Instalación Mejora -TFG-UBUSetas #36 III New Issues ↑Sprint-7	(2)
S7-0.Recoger todos los enlaces de las claves. Eclipse Instalación -TFG-UBUSetas #37 III New Issues ↑Sprint-7	(2)

Figura A.10: Issues del sprint 7.

Sprint 8

Este sprint se desarrolló entre los días 2 y 8 de noviembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Extraer fotografías de las especies que se encuentran en la clave y no en el clasificador, con el fin de ampliar el número de especies recogidas por el clasificador.
- Reentrenar el clasificador con las nuevas imágenes recopiladas.
- Ajustar los métodos de la aplicación Java para que extraiga información de las nuevas especies.

- Añadir nueva información (género, comestibilidad y enlace) de las especies de setas para ser incorporada a la aplicación Android.
- Estudio de las directrices de *material design* para implementar la interfaz de la aplicación Android.
- Estudiar el funcionamiento de la herramienta Latex para empezar la memoria de la documentación.

En esta semana se decidió ampliar el número de especies clasificadas por el clasificador aumentando el número hasta las 171 especies. El objetivo era que no hubiera especies contenidas en la clave dicotómica de géneros que si estuvieran en el clasificador, con el fin de aprovechar la clave conseguida.

Esto provocó la modificación de los métodos en ambas aplicaciones para extraer y manejar las nuevas especies incorporadas.

También se empezó a estudiar cómo realizar una mejor interfaz basándome en los consejos ofrecidos por *material design*.

Se comenzó a estudiar la herramienta latex para empezar lo antes posible con la documentación del proyecto.



Figura A.11: Gráfico Burndown del sprint 8.

Completed Issues	Story points
⑤ S8-1. Incorporación de las fotos de ejemplo de las nuevas especies Android Desarrollo Mejora -TFG-UBUSetas #38 III New Issues ↑ Sprint-8	(1)
⑥ S8-2. Carga en la base de datos de las nuevas especies Desarrollo Eclipse Mejora -TFG-UBUSetas #39 III New Issues ↑ Sprint-8	(2)
⑦ S8-3. Incorporación del género, comestibilidad y link en la actividad mostrar Infor... Android Desarrollo Mejora -TFG-UBUSetas #40 III New Issues ↑ Sprint-8	(2)
⑧ S8-4. Arreglar error de tablaEnlaces Eclipse Error -TFG-UBUSetas #41 III New Issues ↑ Sprint-8	(1)
⑨ S8-4. Estudio de la herramienta latex para elaborar la documentación Documentación Estudio -TFG-UBUSetas #42 III New Issues ↑ Sprint-8	(5)
⑩ S8-5. Descargar imágenes de las especies para probar el clasificador. Test -TFG-UBUSetas #43 III New Issues ↑ Sprint-8	(5)
⑪ S8-6. Estudio de material design Estudio -TFG-UBUSetas #44 III New Issues ↑ Sprint-8	(5)
⑫ S8-7. Modificación de los layout para que se muestren correctamente en horizontal Android Desarrollo Mejora -TFG-UBUSetas #45 III New Issues ↑ Sprint-8	(2)

Figura A.12: Issues del sprint 8.

Sprint 9

Este sprint se desarrolló entre los días 8 y 15 de noviembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Escribir la introducción de la documentación.
- Escribir los objetivos del proyecto de la documentación.
- Escribir los conceptos teóricos de la documentación.
- Escribir las técnicas y herramientas de la documentación.
- Crear el prototipado para la interfaz de la aplicación Android.

Este sprint se dedicó a comenzar la documentación del proyecto y a crear un prototipado de la interfaz de la aplicación Android que sirviera de guía para construir la interfaz de la aplicación Android.



Figura A.13: Gráfico Burndown del sprint 9.

Completed Issues	Story points
S9-1. Subir la plantilla inicial de la documentación en latex Documentación -TFG-UBUSetas #46 III New Issues ↑Sprint-9	①
S9-2. Creación de la introducción del proyecto Documentación -TFG-UBUSetas #47 III New Issues ↑Sprint-9	①
S9-3. Elaboración de los objetivos del proyecto Documentación -TFG-UBUSetas #48 III New Issues ↑Sprint-9	①
S9-4. Elaboración de los conceptos teóricos Documentación -TFG-UBUSetas #49 III New Issues ↑Sprint-9	⑧
S9-5. Explicación de las técnicas y herramientas Documentación -TFG-UBUSetas #50 III New Issues ↑Sprint-9	⑤
S9-6. Creación del prototipado de la interfaz de android Android Documentación -TFG-UBUSetas #51 III New Issues ↑Sprint-9	③

Figura A.14: Issues del sprint 9.

Sprint 10

Este sprint se desarrolló entre los días 15 y 22 de noviembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Construir la interfaz de las actividades creadas hasta este punto.
- Solucionar errores por los que las imágenes no se muestran correctamente en la aplicación Android.
- Seguir desarrollando la documentación.

- Creación de las actividades que muestran un listado de las especies y claves dicotómicas disponibles en la aplicación.

Esta semana se dedicó a seguir construyendo la aplicación Android, incorporando dos nuevas actividades que mostraran al usuario las claves dicotómicas e información de las diferentes especies disponibles.

Se corrigió un error por el que el tamaño de la foto insertada por el usuario afectaba en los resultados ofrecidos por el clasificador, ya que se debían proporcionar imágenes escaladas a 224x224 pixeles de tamaño al clasificador para un correcto funcionamiento.

Se revisó la documentación creada en el sprint anterior.



Figura A.15: Gráfico Burndown del sprint 10.

Completed Issues	Story points
S10-1. Creación de la actividad lanzadora de la aplicación Android Desarrollo Primera Versión -TFG-UBUSetas #52 III New Issues ↑Sprint-10	(2)
S10-2. Cambiar la actividad recoger foto Android Desarrollo Mejora -TFG-UBUSetas #53 III New Issues ↑Sprint-10	(4)
S10-3. Creación de la actividad mostrar claves Android Desarrollo Primera Versión -TFG-UBUSetas #54 III New Issues ↑Sprint-10	(4)
S10-4. Creación de la actividad mostrar setas Android Desarrollo Primera Versión -TFG-UBUSetas #55 III New Issues ↑Sprint-10	(4)
S10-6. Modificar la actividad clave dicotómica, interfaz Android Desarrollo Mejora -TFG-UBUSetas #57 III New Issues ↑Sprint-10	(2)

Figura A.16: Issues del sprint 10.

Sprint 11

Este sprint se desarrolló entre los días 22 y 30 de noviembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Seguir desarrollando la interfaz Android.
- Crear una actividad que pida al usuario elegir sobre qué géneros, de los clasificados, desea aplicar la clave dicotómica para concretar las preguntas realizadas en esos géneros.
- Seguir desarrollando la documentación.
- Corregir pequeños errores de funcionamiento de la interfaz.
- Generar la primera *release* del proyecto.

En este sprint se siguió implementando la aplicación Android añadiendo nuevas funcionalidades, como el filtrado de géneros de la clave dicotómica. Así como corregir pequeños errores que se encontraron en la interfaz.

Tras realizar estas tareas se lanzó la primera *release* del proyecto.

Se terminaron los puntos de la memoria que no habían sido completados todavía.

A.2. PLANIFICACIÓN TEMPORAL

17



Figura A.17: Gráfico Burndown del sprint 11.

Completed Issues	Story points
⌚ S11-4. Modificación de la actividad clave dicotómica Android Desarrollo Primera Versión -TFG-UBUSetas #34 III New Issues ↑Sprint-11	(5)
⌚ S11-1. Modificación de la interfaz de la actividad mostrar resultados Android Desarrollo Mejora -TFG-UBUSetas #58 III New Issues ↑Sprint-11	(2)
⌚ S11-2. Modificar la actividad mostrar comparativa Android Desarrollo Mejora -TFG-UBUSetas #59 III New Issues ↑Sprint-11	(1)
⌚ S11-3. Creacion de la actividad elegir clave Android Desarrollo Primera Versión -TFG-UBUSetas #60 III New Issues ↑Sprint-11	(5)
⌚ S11-4. Terminar de documentar los códigos de la aplicación móvil Android Documentación -TFG-UBUSetas #61 III New Issues ↑Sprint-11	(3)
⌚ S11-5. Corregir el error de la cámara de fotos. Android Error -TFG-UBUSetas #62 III New Issues ↑Sprint-11	(2)
⌚ S11-7. Terminar el punto 4 de la documentación. Documentación -TFG-UBUSetas #64 III New Issues ↑Sprint-11	(2)
⌚ S11-8. Revisar la parte realizada de la documentación. Documentación -TFG-UBUSetas #65 III New Issues ↑Sprint-11	(3)
⌚ S11-9. Incorporar los botones flotantes en la actividad lanzadora. Android Desarrollo Mejora -TFG-UBUSetas #66 III New Issues ↑Sprint-11	(2)

Figura A.18: Issues del sprint 11.

Sprint 12

Este sprint se desarrolló entre el día 30 de noviembre al 7 de diciembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Traducir todos los textos de la aplicación, claves e información al inglés.

- Internacionalizar la aplicación permitiendo que el usuario pueda elegir entre el español y el inglés, pudiendo cambiar en cualquier momento.
- Modificar el proyecto de Eclipse para que se traduzcan automáticamente todas las claves dicotómicas.
- Incorporar botones que muestren ayuda dentro de la aplicación al usuario en todas las actividades.
- Corregir errores en la rotación de pantalla de la aplicación Android.

Esta semana se dedicó a traducir todos los textos de la aplicación Android al inglés, lo que provocó que se necesitara modificar el método extractor de claves dicotómicas del proyecto de Eclipse para que a la vez que extrajera las claves, las tradujera al inglés.

Se añadieron páginas de ayuda de usuario, dentro de la aplicación Android, para explicar la funcionalidad de cada elemento que se muestra al usuario en pantalla. Para acceder a ella, el usuario solo debe pulsar el botón de ayuda, o el ítem de ayuda disponible en el menú de la aplicación.



Figura A.19: Gráfico Burndown del sprint 12.

Completed Issues	Story points
⌚ S10-5. Desarrollo del punto 5 de la documentación. Documentación -TFG-UBUSetas #56 III New Issues ↑Sprint-12	(2)
⌚ S11-10. Internacionalizar los textos de la aplicación. Android Desarrollo Mejora -TFG-UBUSetas #67 III New Issues ↑Sprint-12	(4)
⌚ S12-1. Corregir error en la elección de géneros para la clave general. Android Error -TFG-UBUSetas #68 III New Issues ↑Sprint-12	(1)
⌚ S12-2. Lanzar la primera release de la aplicación en github. Android Desarrollo Primera Versión -TFG-UBUSetas #69 III New Issues ↑Sprint-12	(2)
⌚ S12-3. Arreglar error por el que los botones desaparecen Android Desarrollo Error -TFG-UBUSetas #70 III New Issues ↑Sprint-12	(1)
⌚ S12-4. Generar ayuda de la actividad lanzadora Android Desarrollo Mejora -TFG-UBUSetas #71 III New Issues ↑Sprint-12	(2)
⌚ S12-5. Arreglar error por el que el idioma cambia Android Error -TFG-UBUSetas #72 III New Issues ↑Sprint-12	(4)
⌚ S12-6. Arreglar error en mostrar información para que se muestre correctamente e... Android Error -TFG-UBUSetas #73 III New Issues ↑Sprint-12	(1)
⌚ S12-7. Traducir las claves dicotómicas al inglés. Desarrollo Eclipse Mejora -TFG-UBUSetas #74 III New Issues ↑Sprint-12	(4)
⌚ S12-8. Arreglar error de traducción. Android Error -TFG-UBUSetas #75 III New Issues ↑Sprint-12	(3)
⌚ S12-9. Generar ayuda del menú lateral. Android Desarrollo -TFG-UBUSetas #76 III New Issues ↑Sprint-12	(1)
⌚ S12-10. Generar ayuda de la actividad recoger foto Android Desarrollo -TFG-UBUSetas #77 III New Issues ↑Sprint-12	(1)
⌚ S12-11. Generar ayuda de la actividad mostrar claves. Android Desarrollo -TFG-UBUSetas #78 III New Issues ↑Sprint-12	(1)
⌚ S12-12. Generar ayuda de la actividad mostrar setas. Android Desarrollo -TFG-UBUSetas #79 III New Issues ↑Sprint-12	(1)
⌚ S12-13. Generar ayuda de la actividad mostrar resultados. Android Desarrollo -TFG-UBUSetas #80 III New Issues ↑Sprint-12	(1)
⌚ S12-14. Generar ayuda de la actividad mostrar clave dicotómica. Android Desarrollo -TFG-UBUSetas #81 III New Issues ↑Sprint-12	(1)
⌚ S12-15. Generar ayuda de la actividad mostrar información. Android Desarrollo -TFG-UBUSetas #82 III New Issues ↑Sprint-12	(1)
⌚ S12-16. Generar ayuda de la actividad elegir clave. Android Desarrollo -TFG-UBUSetas #83 III New Issues ↑Sprint-12	(1)
⌚ S12-17. Generar ayuda de la actividad mostrar comparativa. Android Desarrollo -TFG-UBUSetas #84 III New Issues ↑Sprint-12	(1)

Figura A.20: Issues del sprint 12.

Sprint 13

Este sprint se desarrolló entre los días 7 y 14 de diciembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Estudiar la herramienta *Roboelectric* para realizar los test unitarios en Android Studio.
- Estudiar las herramientas *Espresso* y *UIautomator* para realizar las pruebas de integración en Android Studio.

- Realizar los primeros test unitarios de la aplicación.
- Incorporar botones que muestren ayuda dentro de la aplicación al usuario en todas las actividades.
- Escribir el punto de *Aspectos relevantes del desarrollo del proyecto* de la documentación.
- Escribir el punto de *Trabajos relacionados* de la documentación.
- Escribir el punto de *Conclusiones y Líneas de trabajo futuras* de la documentación.

Este sprint se dedicó a estudiar cómo realizar los test unitarios y de integración a la aplicación Android.

Se elaboraron los primeros test unitarios y se siguió avanzando en el desarrollo de la documentación.



Figura A.21: Gráfico Burndown del sprint 13.

Completed Issues	Story points
⌚ S13-1. Terminar el apartado 5 de la documentación. Documentación -TFG-UBUSetas #85 III New Issues ↑Sprint-13	(4)
⌚ S13-2. Realizar el apartado 6 de la documentación. Documentación -TFG-UBUSetas #86 III New Issues ↑Sprint-13	(2)
⌚ S13-3. Estudiar herramientas de testeo para Android. Android Estudio Test -TFG-UBUSetas #87 III New Issues ↑Sprint-13	(4)
⌚ S13-4. Generar los test unitarios de la actividad lanzadora. Android Test -TFG-UBUSetas #88 III New Issues ↑Sprint-13	(3)
⌚ S13-5. Estudiar la herramienta Uiautomator para Android Android Estudio -TFG-UBUSetas #89 III New Issues ↑Sprint-13	(3)
⌚ S13-6. Corrección de las formulas de Latex y añadir el punto Data Augmentation a... Documentación -TFG-UBUSetas #90 III New Issues ↑Sprint-13	(2)
⌚ S13-7. Elaboración del apartado 7 de la documentación. Documentación -TFG-UBUSetas #91 III New Issues ↑Sprint-13	(2)
⌚ S13-8. Crear las pruebas unitarias de la actividad mostrar resultados. Android Test -TFG-UBUSetas #92 III New Issues ↑Sprint-13	(2)

Figura A.22: Issues del sprint 13.

Sprint 14

Este sprint se desarrolló entre los días 14 y 21 de diciembre de 2017. Se realizaron las siguientes tareas y objetivos:

- Realizar los test unitarios de todas las actividades de la aplicación Android.
- Realizar todos los test de integración de la aplicación Android.
- Estudiar la herramienta *monkeyrunner* para crear test de rendimiento.

Este sprint se dedicó a realizar pruebas unitarias y de integración sobre todas las actividades de la aplicación Android.

También se estudió la herramienta *monkeyrunner* para crear una serie de eventos aleatorios sobre la aplicación y comprobar la robustez de esta.



Figura A.23: Gráfico Burndown del sprint 14.

A.2. PLANIFICACIÓN TEMPORAL

23

Completed Issues	Story points
⌚ S14-1. Añadir los test del menú lateral a la actividad mostrar resultados. Android Test -TFG-UBUSetas #93 III New Issues ↑Sprint-14	①
⌚ S14-2. Realizar los test unitarios de la actividad mostrar setas Android Test -TFG-UBUSetas #94 III New Issues ↑Sprint-14	②
⌚ S14-3. Realizar los test unitarios de la actividad mostrar claves Android Test -TFG-UBUSetas #95 III New Issues ↑Sprint-14	②
⌚ S14-4. Realizar los test unitarios sobre la actividad Recoger foto. Android Test -TFG-UBUSetas #96 III New Issues ↑Sprint-14	②
⌚ S14-5. Realizar test unitarios sobre la actividad clave dicotómica. Android Test -TFG-UBUSetas #97 III New Issues ↑Sprint-14	②
⌚ S14-6. Realizar los test unitarios sobre la actividad mostrar información seta. Android Test -TFG-UBUSetas #98 III New Issues ↑Sprint-14	②
⌚ S14-7. Realizar los test unitarios de la actividad elegir claves. Android Test -TFG-UBUSetas #99 III New Issues ↑Sprint-14	②
⌚ S14-8. Realizar los test unitarios de la actividad Mostrar Comparativa Android Test -TFG-UBUSetas #100 III New Issues ↑Sprint-14	②
⌚ S14-9. Crear los test unitarios de la actividad Mostrar Comparativa Android Test -TFG-UBUSetas #101 III New Issues ↑Sprint-14	②
⌚ S14-10. Crear los test de integración de la actividad lanzadora. Android Test -TFG-UBUSetas #102 III New Issues ↑Sprint-14	②
⌚ S14-11. Realizar test de integración de la actividad recoger foto. Android Test -TFG-UBUSetas #103 III New Issues ↑Sprint-14	②
⌚ S14-12. Crear los test de integración de la actividad mostrar setas Android Test -TFG-UBUSetas #104 III New Issues ↑Sprint-14	②
⌚ S14-13. Crear los test de integración de la actividad mostrar claves Android Test -TFG-UBUSetas #105 III New Issues ↑Sprint-14	②
⌚ S14-14. Crear los test de integración de la actividad mostrar resultados. Android Test -TFG-UBUSetas #106 III New Issues ↑Sprint-14	②
⌚ S14-15. Crear los test de integración de la actividad mostrarComparativa Android Test -TFG-UBUSetas #107 III New Issues ↑Sprint-14	②
⌚ S14-16. Crear los test de integración de la actividad Elegir claves. Android Test -TFG-UBUSetas #108 III New Issues ↑Sprint-14	②
⌚ S14-17. Crear los test de integración de la actividad mostrar Información setas. Android Test -TFG-UBUSetas #109 III New Issues ↑Sprint-14	②
⌚ S14-18. Crear los test de integración de la actividad clave dicotómica Android Test -TFG-UBUSetas #110 III New Issues ↑Sprint-14	②
⌚ S14-19. Reestructurar el proyecto de Eclipse. Eclipse Mejora -TFG-UBUSetas #111 III New Issues ↑Sprint-14	①

Figura A.24: Issues del sprint 14.

A.3. Estudio de viabilidad

En este apartado se va a hacer un estudio sobre la viabilidad económica y legal del proyecto.

Viabilidad económica

En esta sección se va a hacer una aproximación de los costes que supondría para una empresa el desarrollo de este proyecto

Coste del personal

La realización de este proyecto ha necesitado de cuatro meses en los que estaría empleado un desarrollador. Vamos a suponer que este desarrollador tiene un salario bruto de 1500 euros.

Ahora habrá que calcular los gastos de seguridad social¹, tanto por parte de la empresa como del trabajador:

Por parte de la empresa:

- Contingencias comunes: 23,6 %
- Desempleo: 5,5 %
- Fogasa: 0,2 %
- Formación profesional: 0,6 %

Por parte del trabajador:

- Contingencias comunes: 4,7 %
- Desempleo: 1,55 %
- Formación profesional: 0,1 %

Hacen un total del 29,9 % por parte de la empresa y el 6,35 % por parte del trabajador.

¹http://www.seg-social.es/Internet_1/Trabajadores/CotizacionRecaudaci10777/Basesytiposdecotiza36537/index.htm

Concepto	Coste
Salario mensual neto	1.224,75
Retención IRPF (12 %)	180
Seguridad social (Empleado) (6,35 %)	95,25
Salario mensual bruto	1.500
Salario total (4 meses)	6.000
Seguridad social (Empresa) (29,9 %)	448,5
Coste total mensual de la empresa	1794

Tabla A.1: Costes de personal

Coste informáticos

A nivel de software no hay gastos ya que todas las librerías son gratuitas y no necesitamos pagar ningún servidor para mantener nuestra aplicación. A nivel hardware solamente se ha usado mi ordenador portátil por valor de 700 euros. La amortización es de 5 años y se ha usado 4 meses por lo que el coste es el siguiente:

$$(700 \text{ euros} / (12 \text{ meses} * 4 \text{ periodos})) * 4 \text{ meses} = 58,33 \text{ euros}$$

Coste total

En la siguiente tabla se recoge el coste total del proyecto.

Concepto	Coste
Salario empleado	6000
Seguridad social	1794
Costes hardware(6,35 %)	58,33
Total	7852,33

Tabla A.2: Costes totales

Viabilidad legal

En esta sección se recogerá una tabla con las distintas licencias de las librerías usadas en el proyecto.

Librería	Licencia
Tensorflow	Apache 2.0 opensource license
Apache jena	Apache License 2.0
Json	Libre
Jaunt	Libre mensualmente
SQlite	Libre

Tabla A.3: Licencias

Respecto a la licencia de Jaunt, hay que mencionar que es una librería que hay que descargar cada mes, ya que se actualiza y deja de funcionar pasado ese tiempo.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apartado se van a detallar los requisitos globales del proyecto presentado, así como los diferentes casos de uso y requisitos funcionales implementados en la aplicación.

B.2. Objetivos generales

A continuación se muestran los objetivos generales del proyecto:

- Investigar las bibliotecas de Tensorflow y los modelos Mobilenet e Inception para crear clasificadores de imágenes que sean capaces de ejecutarse en arquitecturas móviles.
- Investigar técnicas de uso de la Web Semántica para recopilar información de las diferentes especies de setas, de forma automática.
- Investigar técnicas de Web Scraping para extraer gran cantidad de información de una página Web.
- Implementar una aplicación Android que a partir de un clasificador de imágenes y la realización de preguntas al usuario, determine la especie de una seta. Esta aplicación mostrará información de cada especie y filtrará las preguntas de la clave dicotómica de acuerdo a las especies devueltas por el clasificador.

- Generar una aplicación Java que extraiga la información de las especies y las claves dicotómicas de forma automática.
- Internacionalizar la aplicación Android de manera que se pueda visualizar tanto en Inglés como en Español.

B.3. Catalogo de requisitos

En esta sección se van a detallar los diferentes requisitos funcionales implementados en las diferentes aplicaciones y herramientas.

- **RF.1** Crear una herramienta que nos permita descargar información de las diferentes especies de setas.
 - **RF.1.1** Se introducirá un listado de las especies deseadas y la aplicación devolverá la información extraída de la DBpedia.
 - **RF.1.2** La información se podrá descargar tanto en español como en inglés.
- **RF.2** Crear una herramienta que nos permita descargar las diferentes claves dicotómicas.
 - **RF.2.1** La aplicación devolverá una clave dicotómica que discrimine entre géneros de setas.
 - **RF.2.2** La aplicación devolverá claves dicotómicas que discriminan entre las especies de diferentes géneros de setas.
 - **RF.2.3** Las claves se extraerán en español y se traducirán automáticamente al inglés.
 - **RF.2.4** Las claves se serializarán en estructuras de datos java para incorporarlas a la aplicación Android.
- **RF.3** Generar una aplicación para crear una base de datos SQLite para almacenar la información descargada y exportarla a la aplicación Android. Permitirá las siguientes acciones:
 - **RF.3.1** Almacenar la descripción de la especie en Español.
 - **RF.3.2** Almacenar la descripción de la especie en Inglés.
 - **RF.3.3** Almacenar la comestibilidad de la especie en Español.
 - **RF.3.4** Almacenar la comestibilidad de la especie en Inglés.

- **RF.3.5** Almacenar el género de la especie.
 - **RF.3.6** Almacenar un enlace que redirija a la página web de la seta en Wikipedia.
- **RF.4** Usar los algoritmos de Python proporcionados en los ejemplos de Tensorflow para entrenar los clasificadores de imágenes. https://github.com/tensorflow/tensorflow/tree/master/tensorflow/examples/image_retraining
 - **RF.4.1** Elegir el modelo de Mobilenet o Inception.
 - **RF.4.2** Incorporar las imágenes de las especies de setas sobre las que se quiere entrenar el clasificador.
 - **RF.4.3** Determinar el porcentaje de imágenes que van a ser recortadas para crear nuevas.
 - **RF.4.4** Determinar el porcentaje de imágenes que van a ser escaladas para crear nuevas.
 - **RF.4.5** Determinar el porcentaje de imágenes que se va a rotar para crear nuevas.
 - **RF.4.6** Determinar el porcentaje de imágenes sobre las que se va a modificar el brillo para crear nuevas.
 - **RF.4.7** Elegir el número de pasos que va a realizar el clasificador para entrenar el modelo.
- **RF.5** Generar una aplicación Android que permita la clasificación de la especie de una seta con las siguientes funcionalidades:
 - **RF.5.1** El usuario podrá introducir una imagen de la seta desde la cámara del móvil para clasificar.
 - **RF.5.2** El usuario podrá guardar la imagen capturada desde la cámara.
 - **RF.5.3** El usuario podrá introducir una imagen de la seta desde la galería del móvil para clasificar.
 - **RF.5.4** La aplicación mostrará las especies más probables clasificadas para la imagen introducida.
 - **RF.5.5** Se mostrará información e imágenes de ejemplo para cada especie obtenida como resultado.
 - **RF.5.6** El usuario podrá comparar su imagen con las proporcionadas por la aplicación

- **RF.5.7** La aplicación realizará preguntas al usuario para clasificar la especie y reforzar la tarea de clasificación.
- **RF.5.8** Se podrán filtrar las preguntas realizadas para que solo se realicen sobre las especies deseadas.
- **RF.5.9** El usuario podrá cambiar el idioma de la aplicación entre español e inglés.
- **RF.5.10** El usuario podrá acceder a la información de todas las especies a través de un listado de estas.
- **RF.5.11** El usuario podrá acceder a las claves dicotómicas de las especies a través de un listado de estas.
- **RF.5.12** La aplicación mostrará páginas de ayuda que guíen al usuario dentro de la aplicación.

B.4. Especificación de requisitos

Diagramas de casos de uso

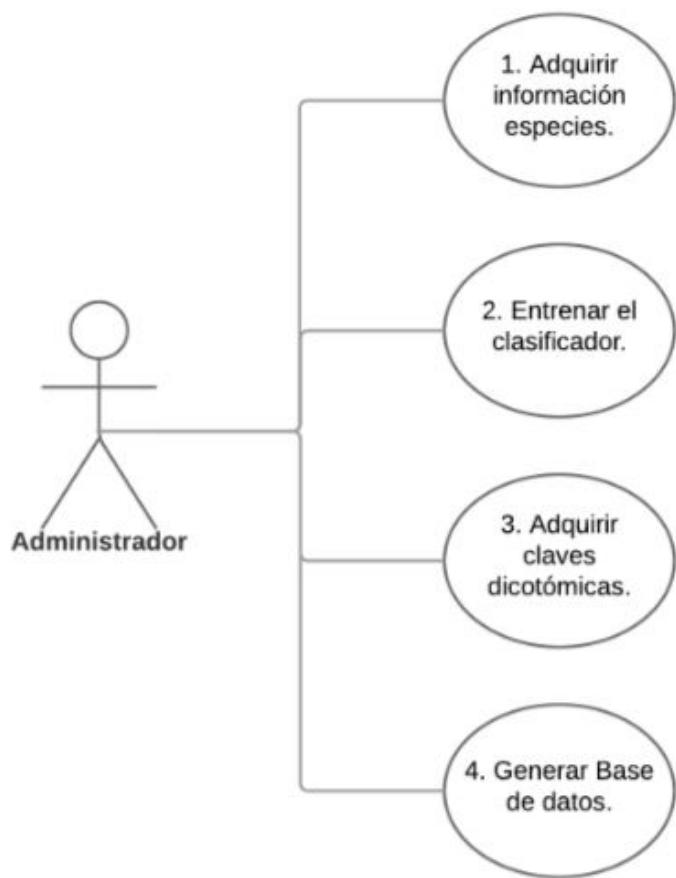


Figura B.1: Diagrama de casos de uso del administrador.

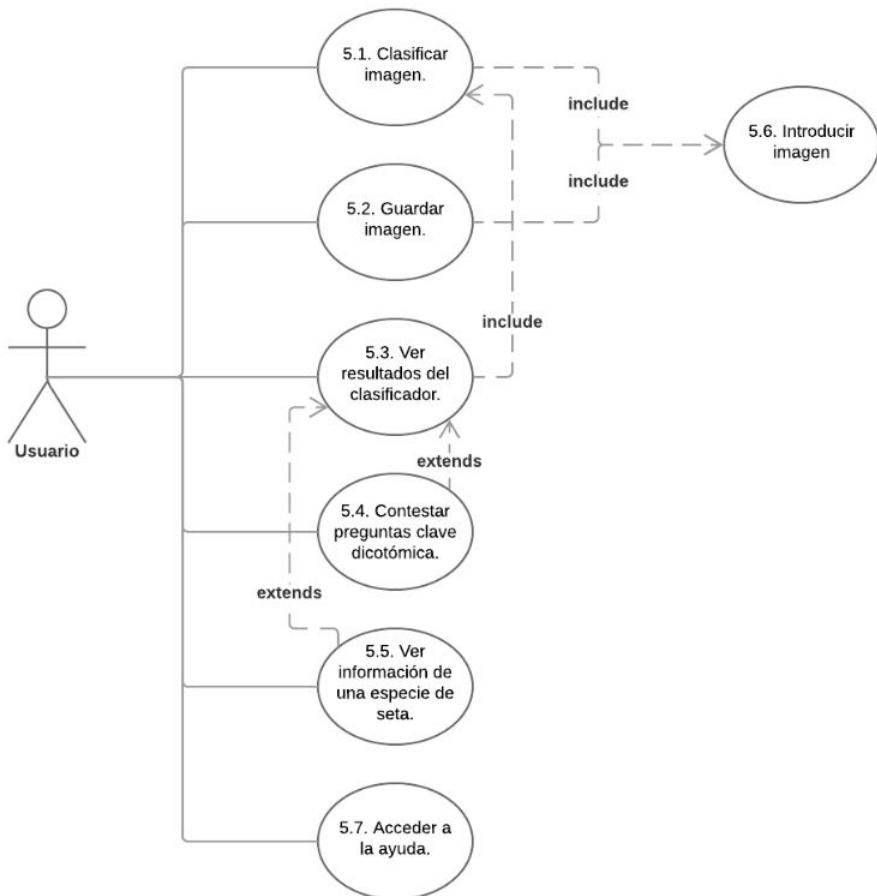


Figura B.2: Diagrama de casos de uso del usuario.

Especificación de los casos de uso

Caso de uso 1: Adquirir información especies.

Descripción	Permite al administrador extraer la información de la especies de la DBpedia.	
Requisitos	RF-1 RF-1.1 RF-1.2	
Precondiciones	Se necesita conexión a Internet para acceder a la DBpedia y tener creadas las tablas en la base de datos SQLite.	
Secuencia normal	Paso	Acción
	1	El administrador indica la ruta de la base de datos.
	2	Se inserta un listado de las especies a descargar.
	3	Se inicia el programa
	4	Se carga la base de datos con la información extraída de la DBpedia.
Postcondiciones	Se ha descargado la información de la especies de setas.	
Excepciones	No se ha podido acceder a la DBpedia.	
Importancia	Media	
Urgencia	Media	

Caso de uso 2: Entrenar el clasificador.													
Descripción	Permite al administrador reentrenar el clasificador de imágenes.												
Requisitos	RF-4 RF-4.1 RF-4.2 RF-4.3 RF-4.4 RF-4.5 RF-4.6 RF-4.7												
Precondiciones	Tener instalado Tensorflow y Python en el sistema.												
Secuencia normal	<table border="1"> <thead> <tr> <th>Paso</th><th>Acción</th></tr> </thead> <tbody> <tr> <td>1</td><td>Se indica el modelo a reentrenar.</td></tr> <tr> <td>2</td><td>Se incorporan las imágenes con las que se quiere entrenar.</td></tr> <tr> <td>3</td><td>Se ajustan los parámetros para realizar el Data Augmentation.</td></tr> <tr> <td>4</td><td>Se elige el número de pasos que va a realizar el programa para entrenar</td></tr> <tr> <td>5</td><td>Se inicia el programa.</td></tr> </tbody> </table>	Paso	Acción	1	Se indica el modelo a reentrenar.	2	Se incorporan las imágenes con las que se quiere entrenar.	3	Se ajustan los parámetros para realizar el Data Augmentation.	4	Se elige el número de pasos que va a realizar el programa para entrenar	5	Se inicia el programa.
Paso	Acción												
1	Se indica el modelo a reentrenar.												
2	Se incorporan las imágenes con las que se quiere entrenar.												
3	Se ajustan los parámetros para realizar el Data Augmentation.												
4	Se elige el número de pasos que va a realizar el programa para entrenar												
5	Se inicia el programa.												
Postcondiciones	Se ha entrenado un modelo para clasificar especies de setas.												
Excepciones	Número insuficiente de imágenes para entrenar.												
Importancia	Alta												
Urgencia	Alta												

Tabla B.2: Caso de uso 2: Entrenar el clasificador.

Caso de uso 3: Adquirir claves dicotómicas.

Descripción	Permite al administrador extraer las claves de la página Web http://www.avelinosetas.info/claves.php .	
Requisitos	RF-2 RF-2.1 RF-2.2 RF-2.3 RF-2.4	
Precondiciones	Se necesita conexión a Internet para acceder a la página web.	
Secuencia normal	Paso	Acción
	1	El administrador indica el nombre del archivo donde se quiere exportar las claves.
	2	Se inserta un listado de las claves a descargar.
	3	Se inicia el programa
	4	Se descargan las claves y se serializan en el archivo indicado.
Postcondiciones	Se obtienen las claves en un archivo serializado para ser exportado a la aplicación Android.	
Excepciones	No se ha podido acceder a la página web.	
Importancia	Alta	
Urgencia	Media	

Tabla B.3: Caso de uso 3: Adquirir claves dicotómicas.

Caso de uso 4: Generar Base de datos.									
Descripción	Permite al administrador preparar la base de datos para almacenar la información de las setas.								
Requisitos	RF-1 RF-2								
Precondiciones	Haber creado una base de datos SQLite en el sistema.								
Secuencia normal	<table border="1"> <thead> <tr> <th>Paso</th><th>Acción</th></tr> </thead> <tbody> <tr> <td>1</td><td>El administrador indica la ruta donde se encuentra la base de datos SQLite.</td></tr> <tr> <td>2</td><td>Se inicia el programa.</td></tr> <tr> <td>3</td><td>El programa crea las tablas necesarias en la base de datos.</td></tr> </tbody> </table>	Paso	Acción	1	El administrador indica la ruta donde se encuentra la base de datos SQLite.	2	Se inicia el programa.	3	El programa crea las tablas necesarias en la base de datos.
Paso	Acción								
1	El administrador indica la ruta donde se encuentra la base de datos SQLite.								
2	Se inicia el programa.								
3	El programa crea las tablas necesarias en la base de datos.								
Postcondiciones	Se crean las tablas en la base de datos.								
Excepciones	Excepción SQL.								
Importancia	Media								
Urgencia	Media								

Tabla B.4: Caso de uso 4: Generar Base de datos.

Caso de uso 5.1: Clasificar imagen.

Descripción	Permite al usuario clasificar la especie a la que pertenece la seta que aparezca en la foto introducida.	
Requisitos	RF-5 RF-5.4	
Precondiciones	Haber cargado la imagen a clasificar.	
Secuencia normal	Paso	Acción
	1	El usuario pulsa en el botón de clasificar, bien desde la actividad lanzadora o desde el menú.
	2	El usuario carga la imagen deseada.
	3	El usuario pulsa sobre el botón de clasificar la imagen.
	4	El sistema muestra las especies más probables clasificadas para esa imagen.
Postcondiciones	Se muestran los resultados obtenidos.	
Excepciones	Error en la carga de la imagen. Error en la clasificación.	
Importancia	Alta	
Urgencia	Alta	

Tabla B.5: Caso de uso 5.1: Clasificar imagen.

Caso de uso 5.2: Guardar imagen.

Descripción	Permite al usuario guardar en el sistema la imagen que haya capturado desde el móvil.										
Requisitos	RF-5 RF-5.2										
Precondiciones	Tener acceso a la cámara del móvil.										
Secuencia normal	<table border="1"> <thead> <tr> <th>Paso</th><th>Acción</th></tr> </thead> <tbody> <tr> <td>1</td><td>El usuario pulsa en el botón de clasificar, bien desde la actividad lanzadora o desde el menú.</td></tr> <tr> <td>2</td><td>El usuario elige la opción de cargar la imagen desde el móvil.</td></tr> <tr> <td>3</td><td>Se introduce la imagen.</td></tr> <tr> <td>4</td><td>El usuario pulsa sobre el botón de guardar.</td></tr> </tbody> </table>	Paso	Acción	1	El usuario pulsa en el botón de clasificar, bien desde la actividad lanzadora o desde el menú.	2	El usuario elige la opción de cargar la imagen desde el móvil.	3	Se introduce la imagen.	4	El usuario pulsa sobre el botón de guardar.
Paso	Acción										
1	El usuario pulsa en el botón de clasificar, bien desde la actividad lanzadora o desde el menú.										
2	El usuario elige la opción de cargar la imagen desde el móvil.										
3	Se introduce la imagen.										
4	El usuario pulsa sobre el botón de guardar.										
Postcondiciones	Se almacena la imagen en la galería del móvil.										
Excepciones	Error en la carga de la imagen. Error al adquirir los permisos del móvil.										
Importancia	Baja										
Urgencia	Baja										

Tabla B.6: Caso de uso 5.2: Guardar imagen.

Caso de uso 5.3: Ver resultados del clasificador.

Descripción	Permite al usuario ver los resultados obtenidos de la clasificación y obtener información de las especies.	
Requisitos	RF-5 RF-5.4 RF-5.5 RF-5.6	
Precondiciones		
Secuencia normal	Paso	Acción
	1	El usuario ha clasificado una imagen.
	2	El sistema muestra una lista con los resultados obtenidos.
	3	Si el usuario pulsa una especie de los resultados, se mostrará una imagen de ejemplo que se comparará con la introducida por el usuario.
	4	Si el usuario mantiene pulsada una especie de los resultados, se mostrará información describiendo la especie.
Postcondiciones	Se muestra información de los resultados	
Excepciones	Error en la carga de la imagen. Error al acceder a la base de datos.	
Importancia	Media	
Urgencia	Media	

Tabla B.7: Caso de uso 5.3: Ver resultados del clasificador.

Caso de uso 5.4: Contestar preguntas clave dicotómica.		
Descripción	El sistema realiza una serie de preguntas al usuario con el fin de clasificar el género o especie de la seta.	
Requisitos	RF-5 RF-5.7 RF-5.8	
Precondiciones		
Secuencia después de clasificar una imagen.		
	1	El usuario clasifica una imagen.
	2	Se pulsa el botón de acceder a la clave dicotómica.
	3	El usuario puede elegir sobre qué especies filtrar las preguntas o realizar todas las preguntas.
	4	Una vez seleccionadas las especies, se pulsa sobre el botón de clasificar.
	5	El usuario va respondiendo a las preguntas que le aparecen hasta que se muestra la especie clasificada.
Secuencia desde el listado de claves	Acción	
	1	El usuario pulsa el botón de mostrar claves.
	2	El sistema muestra una lista con las claves disponibles.
	3	El usuario elige una clave.
	4	El usuario va respondiendo a las preguntas que le aparecen hasta que se muestra la especie clasificada.
Postcondiciones	Se clasifica la especie de la seta mediante una clave dicotómica.	
Excepciones		
Importancia	Alta	
Urgencia	Alta	

Tabla B.8: Caso de uso 5.4: Contestar preguntas clave dicotómica.

Caso de uso 5.5: Ver información de una especie de seta.

Descripción	El sistema muestra información de una especie de seta.										
Requisitos	RF-5 RF-5.10 RF-5.5										
Precondiciones											
Secuencia desde los resultados.	<table border="1"> <thead> <tr> <th>Paso</th> <th>Acción</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>El usuario clasifica una imagen.</td> </tr> <tr> <td>2</td> <td>Se mantiene pulsado sobre uno de los resultados.</td> </tr> <tr> <td>3</td> <td>El sistema muestra información de la especie seleccionada.</td> </tr> </tbody> </table>	Paso	Acción	1	El usuario clasifica una imagen.	2	Se mantiene pulsado sobre uno de los resultados.	3	El sistema muestra información de la especie seleccionada.		
Paso	Acción										
1	El usuario clasifica una imagen.										
2	Se mantiene pulsado sobre uno de los resultados.										
3	El sistema muestra información de la especie seleccionada.										
Secuencia desde el listado de especies.	<table border="1"> <thead> <tr> <th>Paso</th> <th>Acción</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>El usuario pulsa el botón de mostrar setas.</td> </tr> <tr> <td>2</td> <td>El sistema muestra una lista con las especies de setas disponibles.</td> </tr> <tr> <td>3</td> <td>El usuario elige una especie.</td> </tr> <tr> <td>4</td> <td>El sistema muestra información de la especie seleccionada.</td> </tr> </tbody> </table>	Paso	Acción	1	El usuario pulsa el botón de mostrar setas.	2	El sistema muestra una lista con las especies de setas disponibles.	3	El usuario elige una especie.	4	El sistema muestra información de la especie seleccionada.
Paso	Acción										
1	El usuario pulsa el botón de mostrar setas.										
2	El sistema muestra una lista con las especies de setas disponibles.										
3	El usuario elige una especie.										
4	El sistema muestra información de la especie seleccionada.										
Postcondiciones	Se muestra información de la especie seleccionada.										
Excepciones											
Importancia	Media										
Urgencia	Media										

Tabla B.9: Caso de uso 5.5: Ver información de una especie de seta.

Caso de uso 5.6: Introducir imagen.	
Descripción	El sistema permite al usuario introducir una imagen desde la cámara del móvil o desde la galería.
Requisitos	RF-5 RF-5.1 RF-5.3
Precondiciones	
Secuencia desde la Paso	Acción
cámara	1 El usuario pulsa sobre el botón de clasificar. 2 El usuario selecciona el botón de la cámara. 3 Se captura la imagen.
Secuencia desde la Paso	Acción
galería	1 El usuario pulsa sobre el botón de clasificar. 2 El usuario selecciona el botón de la galería. 3 Se selecciona la imagen de la galería.
Postcondiciones	El sistema carga la imagen para ser clasificada.
Excepciones	
Importancia	Alta
Urgencia	Alta

Tabla B.10: Caso de uso 5.6: Introducir imagen.

Caso de uso 5.7: Acceder a la ayuda.

Descripción	El sistema permite al usuario acceder a la ayuda de cada actividad desde el menú o desde el botón de ayuda.	
Requisitos	RF-5 RF-5.12	
Precondiciones		
Secuencia normal	Paso	Acción
	1	El usuario pulsa sobre el botón de ayuda del menú o de la actividad actual.
	2	El sistema muestra la ayuda de la actividad o del menú.
Postcondiciones	El sistema muestra la ayuda.	
Excepciones		
Importancia	Media	
Urgencia	Media	

Tabla B.11: Caso de uso 5.7: Acceder a la ayuda.

Apéndice C

Especificación de diseño

C.1. Introducción

En esta sección se van a detallar los diferentes diseños del software que se han desarrollado para implementar el proyecto.

- Diseño de datos: En esta sección se explicará como están implementados los datos y clases desarrolladas tanto en la aplicación Android como en el proyecto de Eclipse.
- Diseño arquitectónico : En esta sección se explicará como están organizados los paquetes de los proyectos y como se interrelacionan entre sí.
- Diseño procedimental: En esta sección se explicarán los procedimientos más relevantes de la aplicación.

Para crear los diferentes diagramas requeridos en los diseños se han utilizado las herramientas *Dia*¹ y *Astah*².

C.2. Diseño de datos

En esta sección se mostrará como están implementados los datos en la base de datos SQLite y a continuación, se mostrarán los diagramas de clases que detallan la estructura de datos seguida en ambos proyectos.

¹<http://dia-installer.de/index.html.es>

²<http://astah.net/>

Tablas de la base de datos

Para guardar la información de las diferentes especies de setas de la aplicación se han creado las tablas de la figura C.1

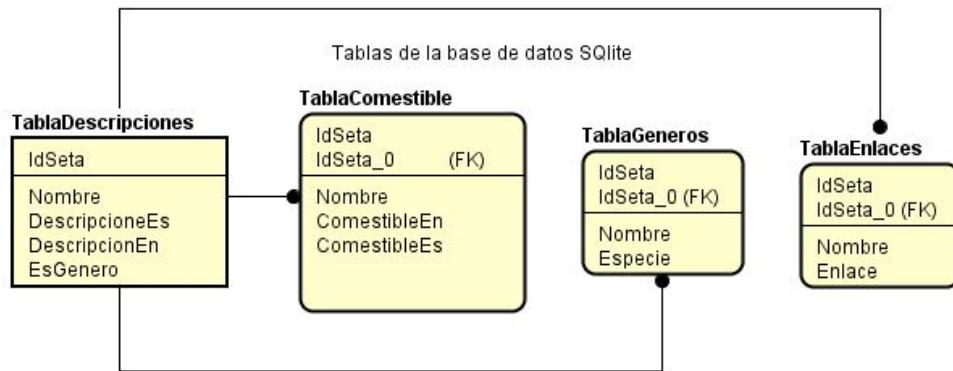


Figura C.1: Tablas de la base de datos SQLite.

- TablaComestible: Almacena la comestibilidad de una determinada especie en español e inglés.
- TablaDescripciones: Almacena la descripción de una determinada especie en español e inglés.
- TablaGeneros: Almacena el género al que pertenece una especie.
- TablaEnlaces: Almacena la url de la *Wikipedia* de la especie.

Almacenamiento claves dicotómicas

Las claves dicotómicas están guardadas en un archivo que contienen las siguientes estructuras java serializadas:

- Map<String, ArrayList <String > >arbolNodos : Este mapa contiene por cada género un arrayList con los siguientes tres mapas:
 - Map<String, ArrayList<String> >arbolNodos: Mapa cuyas claves son los nodos y los valores son los nodos hijos de ese nodo padre.

Este mapa nos sirve para contener la estructura de la clave.

- Map<String, ArrayList<String>>contenidoNodos: Mapa cuyas claves son los nodos y los valores son un array de Strings en el que la primera posición contienen la pregunta de ese nodo, la segunda posición es el enlace de esa especie, la tercera posición es el nodo padre y la cuarta posición contiene el género de esa especie.

Este mapa sirve para almacenar el contenido de cada nodo.

- Map<String, String>generosNodos: Mapa en el que las claves son géneros y los hijos son el nodo que contiene ese género dentro del mapa contenidoNodos.

Este mapa facilita acceder al nodo que contiene el género buscado de seta.

Diagramas de clases

Diagramas de clases del proyecto de Eclipse

En la figura C.2 podemos observar las clases implementadas en el proyecto de Eclipse y que dependencias tienen entre sí. A continuación se muestra una breve descripción de la funcionalidad de cada clase y el paquete en el que esta contenida.

Para acceder a una descripción de los métodos acceder al **javadoc** de cada proyecto.

- Paquete *basedatossql*: Contiene las clases necesarias para crear y manejar la base de datos SQLite.
 - Clase *BDsql*: Clase que contiene los Métodos necesarios para acceder a una base de datos SQLite y manejarla.
 - Clase *CreadorBD*: Clase que crea la base de datos a partir de BDsql y DBpedia.
- Paquete *creador*: Contiene la clase que contiene el *main* del proyecto.
 - Clase *CreadorBDyClaves*: Clase que lanza los métodos necesarios para crear la base de datos y extraer las claves dicotómicas.
- Paquete *dbpedia*: Contiene las clases necesarias para extraer la información de la DBpedia.

- Clase *DBpedia*: Clase que contiene los métodos necesarios realizar consultas a la web semántica DBpedia.
- Paquete *traductor*: Contiene las clases necesarias para implementar un traductor automático.
 - Clase *Translator*: Clase que permite traducir cualquier texto a través de llamadas al traductor de Google. Clase descargada de *archana-testing*.
- Paquete *webscraping*: Contiene las clases necesarias realizar la extracción de las claves dicotómicas mediante técnicas de *Web Scraping*.
 - Clase *ClaveDicotomica*: Clase que guarda la estructura de una clave dicotómica y tiene los métodos para cargarla de la url proporcionada y acceder a sus elementos. Esta preparada para funcionar con las claves de la página web <http://www.avelinosetas.info>
 - Clase *CreadorClaves*: Clase que contiene los métodos necesarios para serializar las claves dicotómicas.

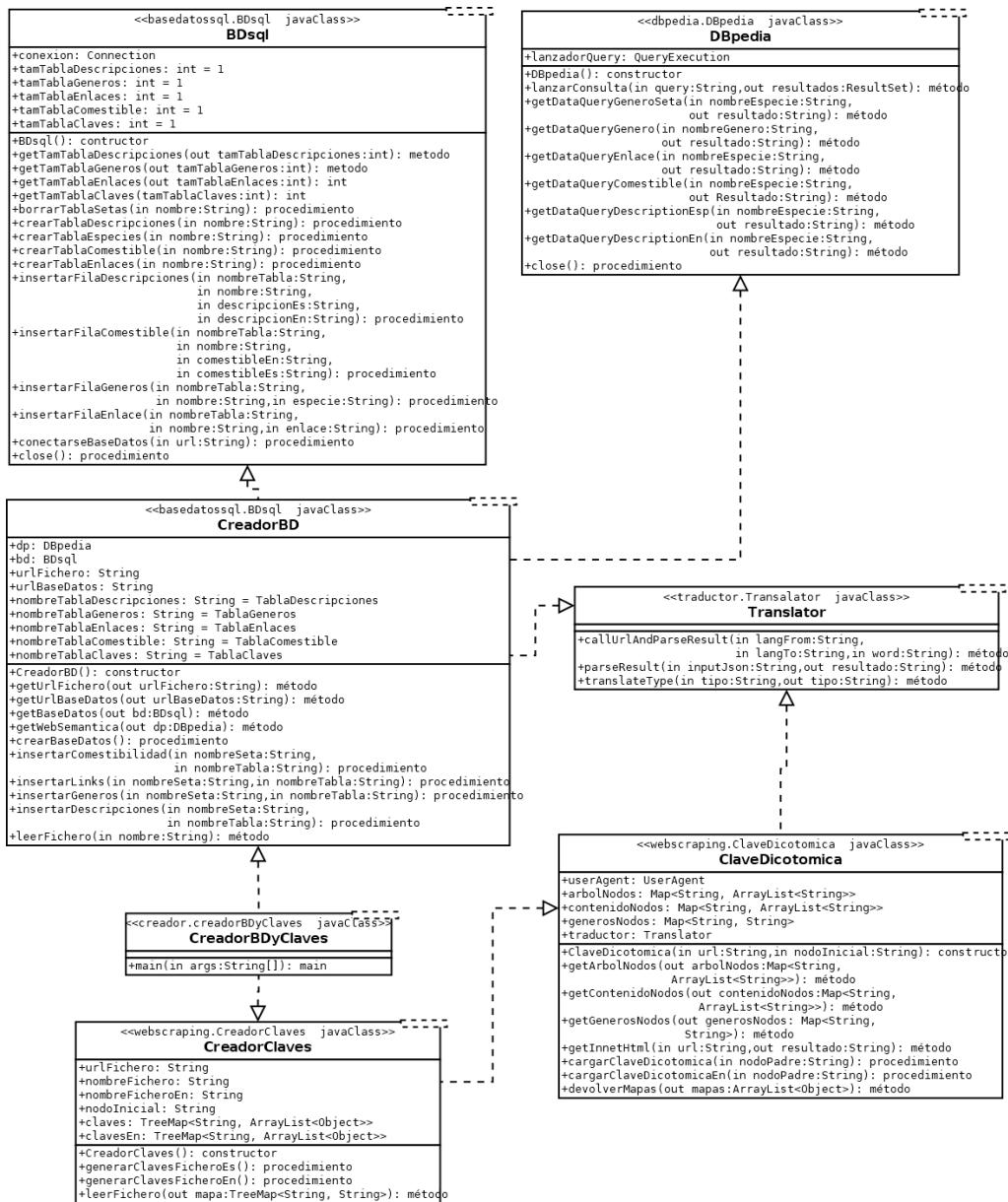


Figura C.2: Diagrama de clases del proyecto de Eclipse.

Diagramas de clases del proyecto Android

En esta sección se van a detallar los diagramas de clases del proyecto Android.

Para acceder a una descripción de los métodos acceder al **javadoc** de

cada proyecto.

- Paquete *basedatos*: Paquete que contiene todas las clases necesarias para acceder a los datos de la aplicación. Diagrama de clases en la figura C.3.
 - Clase *AccesoDatosExternos*: Clase que implementa las funciones necesarias para acceder a los datos externos a la aplicación que se encuentran en la carpeta assets.
 - Clase *DBsetas*: Clase que carga la base de datos SQLite encontrada en assets/databases.
 - Clase *DBsetasManager*: Clase que implementa los métodos para acceder y administrar la base de datos.
- Paquete *clasificador*: Paquete que contiene las clases y actividades relacionadas con la implementación y uso del clasificador de imágenes. Diagrama de clases en la figura C.4.
 - Clase *RecogerFoto*: Clase que implementa la funcionalidad relacionada con la toma y guardado de fotografías.
 - Clase *TensorFlowImageClassifier*: Clase implementada por Tensorflow que permite usar un clasificador entrenado en Android.
- Paquete *clavedicotomica*: Paquete que contiene las clases relacionadas con mostrar las claves dicotómicas. Diagrama de clases en la figura C.5.
 - Clase *ClaveDicotomica*: Clase que implementa la funcionalidad relacionada con mostrar la clave dicotómica seleccionada.
 - Clase *MostrarClaves*: Clase que muestra un listado de las claves dicotómicas de la aplicación.
- Paquete *elegirclaves*: Paquete que contiene las clases relacionadas con el filtrado de géneros de la clave general. Diagrama de clases en la figura C.6.
 - Clase *AdaptadorSelector*: Clase que implementa el adaptador para cargar los elementos (ItemSelector) de la lista de géneros a seleccionar.
 - Clase *ViewHolderSelector*: Clase que implementa los elementos que se deben cargar en la lista (selector) y los relaciona con los elementos de la interfaz.

- Clase *ElegirClaves*: Clase que muestra los géneros a elegir para filtrar la clave dicotómica general.
 - Clase *ItemSelector*: Clase que implementa los elementos del selector.
- Paquete *informacion*: Paquete que contiene las clases relacionadas con mostrar la información de las distintas especies manejadas por la aplicación. Diagrama de clases en la figura C.7.
- Clase *MostrarSetas*: Clase que muestra las setas de la aplicación mediante un listado de tipo RecyclerView.
 - Clase *MostrarInformacionSetas*: Clase que muestra información relativa a la seta pulsada.
- Paquete *lanzador*: Paquete que recoge la clase lanzadora de la aplicación. Diagrama de clases en la figura C.8.
- Clase *Lanzadora*: Clase que arranca la aplicación. Muestra los botones principales para acceder a las funcionalidades más importantes de la aplicación.
- Paquete *resultados*: Paquete que contiene las clases relacionadas con mostrar los resultados obtenidos por el clasificador. Diagrama de clases en la figura C.9.
- Clase *AdaptadorSetasLista*: Clase que sirve de adaptador al sistema para cargar en cada elemento de la lista una foto y el nombre de la especie de la seta.
 - Clase *SetasListaHolder*: Clase que contiene cada par imageView-TextView de cada elemento de la lista.
 - Clase *SetasLista*: Clase que implementa el contenido que va a tener la lista de imágenes que se muestran como resultado tras clasificar una foto.
 - Clase *MostrarComparativa*: Clase que muestra la foto introducida por el usuario y la seleccionada.
 - Clase *MostrarResultados*: Clase que implementa la funcionalidad de la actividad que muestra los resultados obtenidos tras clasificar una foto.
- Paquete *tarjetasclaves*: Paquete que contiene las clases relacionadas con mostrar el listado de claves dicotómicas disponibles en la aplicación. Diagrama de clases en la figura C.10.

- Clase *AdaptadorTarjetasClaves*: Clase que implementa el adaptador para cargar los elementos de la lista de las claves dicotómicas.
 - Clase *ViewHolder*: Clase que implementa los elementos que se deben cargar en la lista de claves.
 - Clase *TarjetaClave*: Clase que implementa el contenido de una tarjeta de claves dicotómicas.
- Paquete *tarjetassetas*: Paquete que contiene las clases relacionadas con mostrar el listado de especies disponibles en la aplicación. Diagrama de clases en la figura [C.11](#).
 - Clase *AdaptadorTarjetasSetas*: Clase que implementa el adaptador para cargar los elementos de la lista de setas.
 - Clase *ViewHolder*: Clase que implementa los elementos que se deben cargar en la lista de setas.
 - Clase *TarjetaSeta*: Clase que implementa el contenido de una tarjeta de la lista de especies de setas.

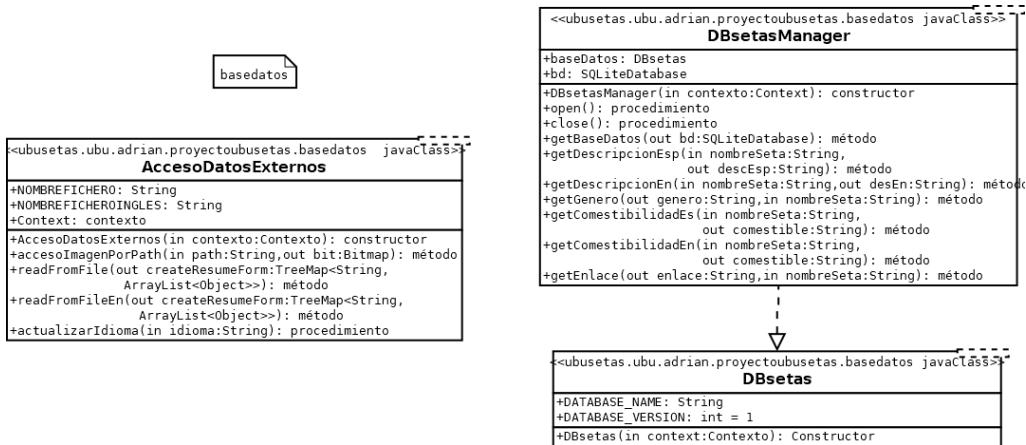


Figura C.3: Diagrama de clases del paquete basedatos.

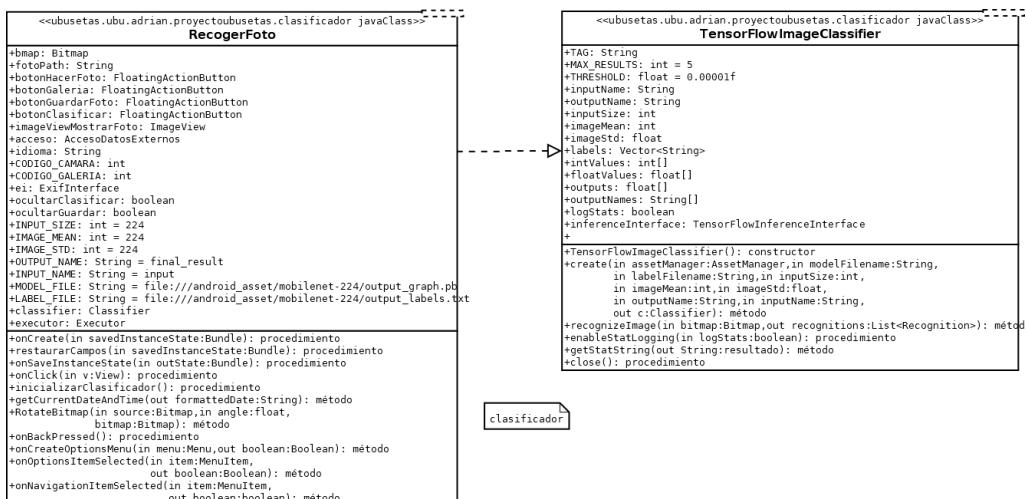


Figura C.4: Diagrama de clases del paquete clasificador.

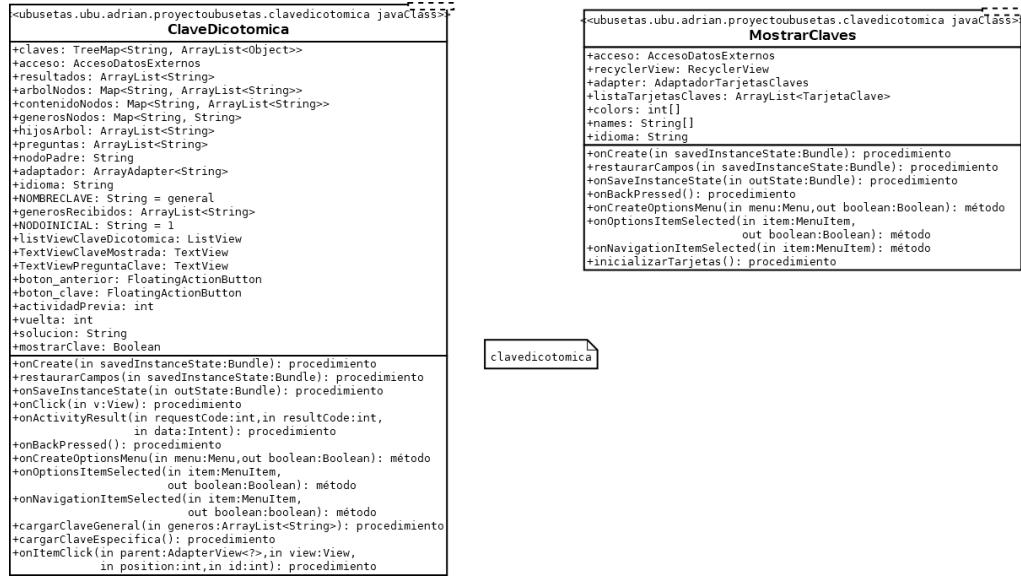


Figura C.5: Diagrama de clases del paquete clavedicotomica.

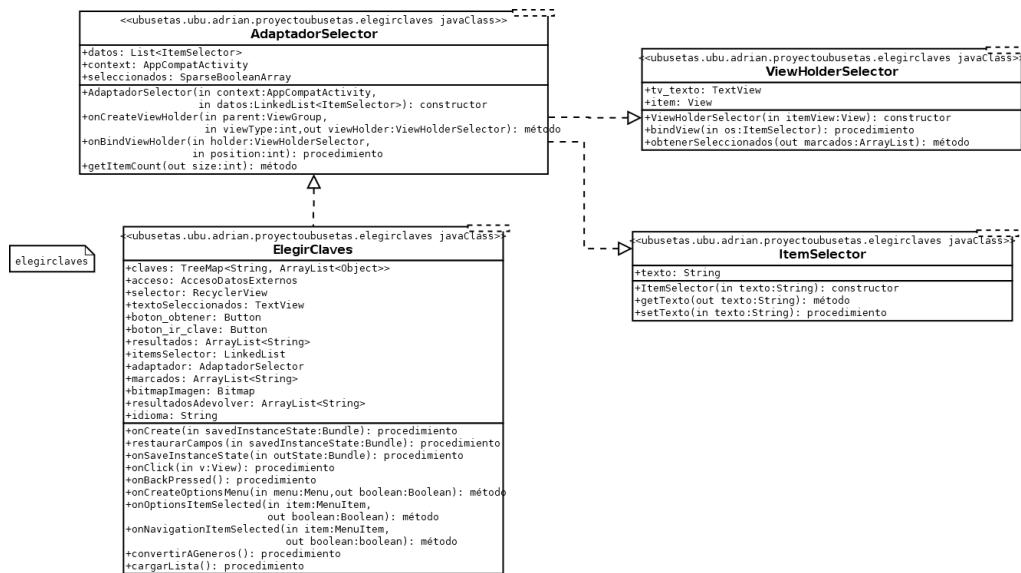


Figura C.6: Diagrama de clases del paquete elegirclaves.

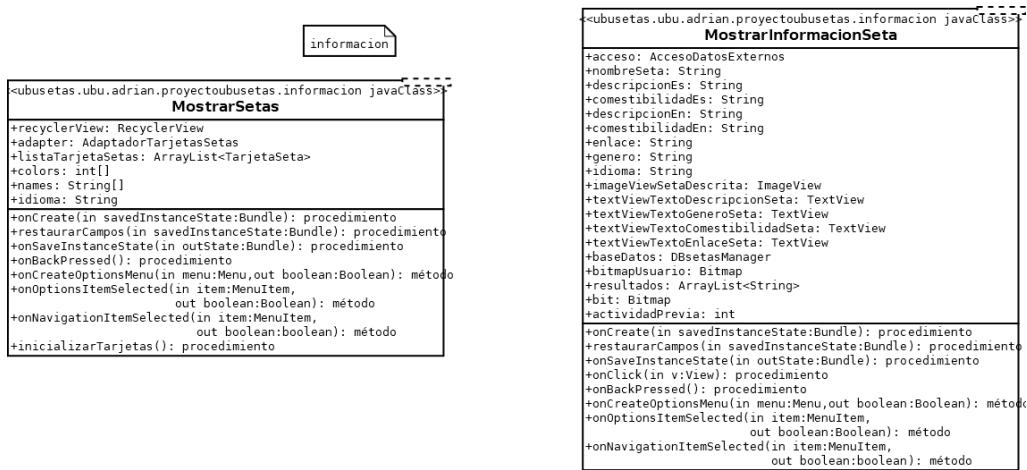


Figura C.7: Diagrama de clases del paquete informacion.

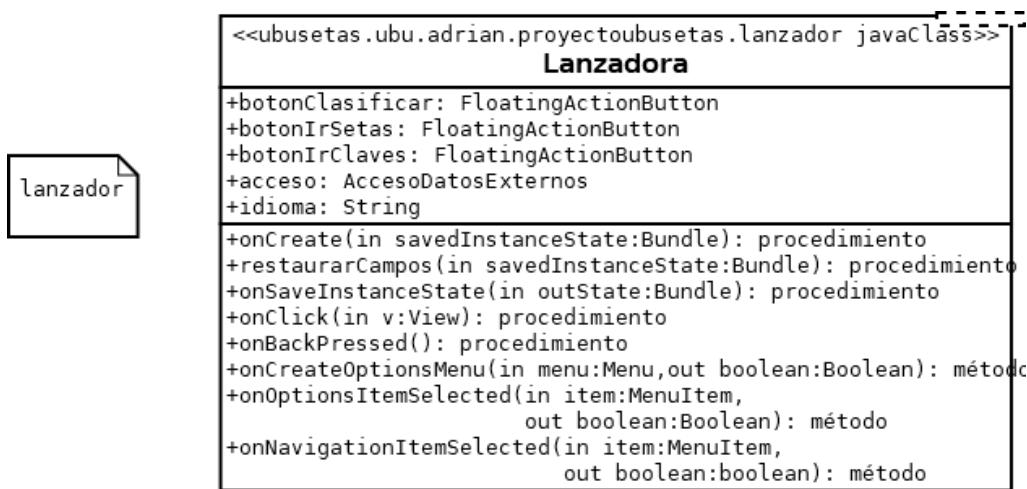


Figura C.8: Diagrama de clases del paquete lanzador.

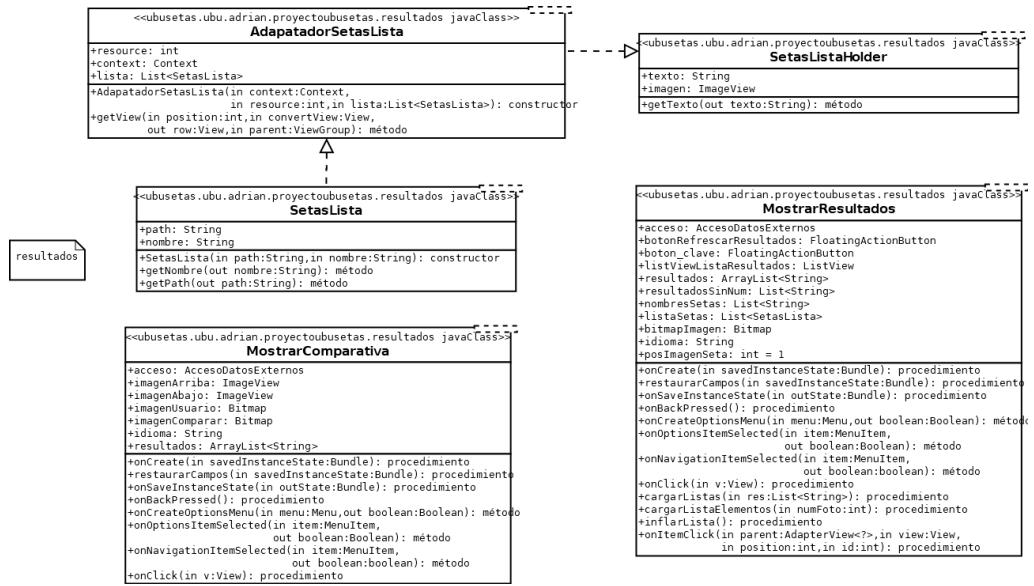


Figura C.9: Diagrama de clases del paquete resultados.

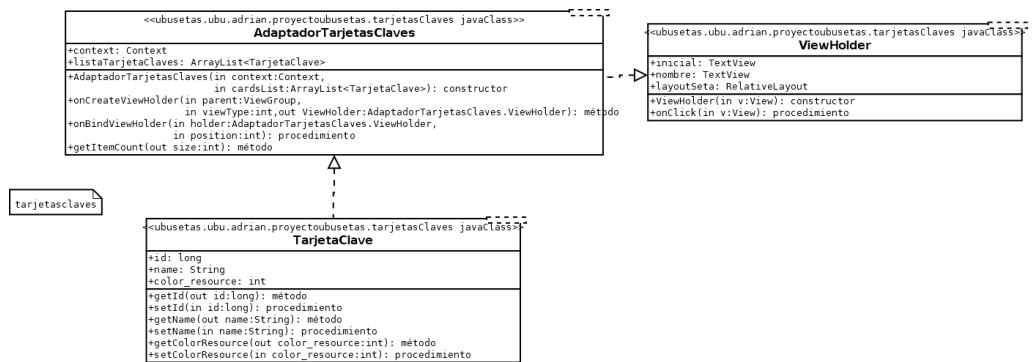


Figura C.10: Diagrama de clases del paquete tarjetasclaves.

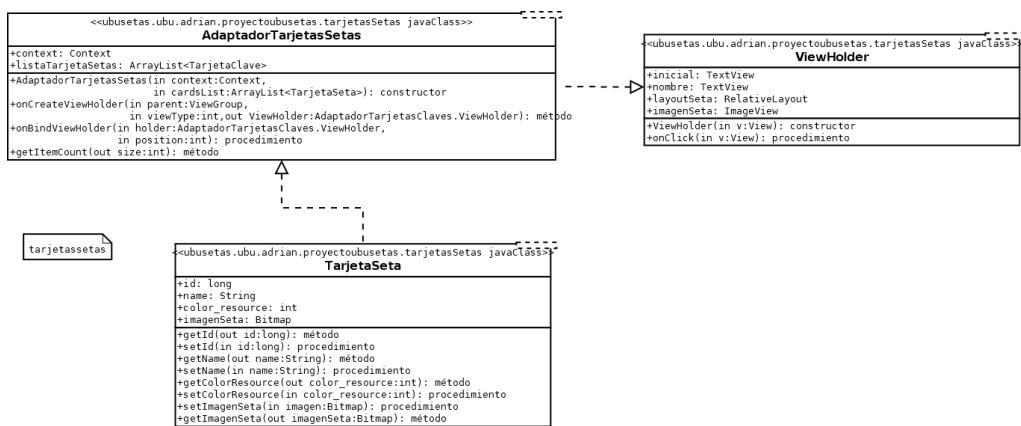


Figura C.11: Diagrama de clases del paquete tarjetassetas.

C.3. Diseño procedimental

Sección en la que se van a explicar los pasos para realizar los procesos más relevantes de la aplicación Android.

Clasificar imagen

Para clasificar una imagen desde la aplicación Andrroi, el usuario deberá seguir estos pasos:

- Paso 1: Iniciar la aplicación.
- Paso 2: Pulsar sobre el botón de *clasificar* disponible en la actividad lanzadora o en el menú.
- Paso 3: Pulsar sobre el botón de galería o cámara disponibles en la siguiente actividad para cargar la imagen, bien desde la galería o desde la cámara de fotos del móvil.
- Paso 4: Introducir una imagen de una seta.
- Paso 5: En este momento, aparecerá el botón de clasificar. Pulsar sobre este botón.
- Paso 6: La aplicación mostrará un listado de las especies más probables clasificadas para esa imagen.

Selección de géneros para usar la clave dicotómica

Una vez el usuario haya clasificado una imagen, podrá acceder a una clave dicotómica filtrada según los resultados obtenidos. El usuario podrá elegir los géneros de setas sobre los que aplicar la clave, si lo desea podrá aplicarla sobre todos los disponibles.

- Paso 1: Realizar los pasos del procedimiento *Clasificar imagen*.
- Paso 2: Pulsar sobre el botón de acceso a la clave dicotómica.
- Paso 3: Seleccionar de la lista los géneros deseados. Para ello primero se eligen los géneros pulsando sobre ellos y después se pulsa sobre el botón de seleccionar.
- Paso 4: Pulsar sobre el botón *ir clave dicotómica*.

- Paso 5: En este momento, se mostrarán las preguntas relacionadas con los géneros seleccionados.

Seleccionar clave dicotómica

El usuario podrá seleccionar una clave de las disponibles en la aplicación.

- Paso 1: Iniciar la aplicación.
- Paso 2: Pulsar sobre el botón de *ir claves* disponible en la actividad lanzadora o en el menú.
- Paso 3: Pulsar sobre la clave dicotómica del género de seta deseado.
- Paso 4: En este momento, se mostrarán las preguntas relacionadas con la clave seleccionada.

Acceder a información de especie de seta

El usuario podrá acceder a la información de una especie de seta recogida por la aplicación. Esta acción se podrá realizar de las siguientes dos formas:

Forma 1

- Paso 1: Iniciar la aplicación.
- Paso 2: Pulsar sobre el botón de *mostrar setas* disponible en la actividad lanzadora o en el menú.
- Paso 3: Pulsar sobre la especie de seta deseada.
- Paso 4: En este momento, se mostrará información describiendo la especie pulsada.

Forma 2

- Paso 1: Realizar los pasos del procedimiento *Clasificar imagen*.
- Paso 2: Mantener pulsado uno de los resultados
- Paso 3: En este momento, se mostrará información describiendo la especie pulsada.

Responder una clave dicotómica

El usuario podrá responder las preguntas de la clave dicotómica. Esta acción se podrá realizar de las siguientes dos formas:

Forma 1

- Paso 1: Realizar los pasos del procedimiento *Seleccionar clave dicotómica*.
- Paso 2: Ir respondiendo a las preguntas que va realizando la aplicación.
- Paso 3: El usuario podrá volver a la pregunta anterior pulsando el botón de volver.
- Paso 4: Una vez se hayan respondido todas las preguntas, se mostrará el género o especie clasificado por la clave, según esta sea la clave de géneros o una de especies.
- Paso 5: Si la aplicación tiene una clave dicotómica del género clasificado, se notificará al usuario y aparecerá un botón que le redirija a la clave que discrimina las especies de ese género.

Forma 2

- Paso 1: Realizar los pasos del procedimiento *Selección de géneros para usar la clave dicotómica*.
- Paso 2: Ir respondiendo a las preguntas que va realizando la aplicación.
- Paso 3: El usuario podrá volver a la pregunta anterior pulsando el botón de volver.
- Paso 4: Una vez se hayan respondido todas las preguntas, se mostrará el género clasificado por la clave
- Paso 5: Si la aplicación tiene una clave dicotómica del género clasificado, se notificará al usuario y aparecerá un botón que le redirija a la clave que discrimina las especies de ese género.

C.4. Diseño arquitectónico

En esta sección se va a mostrar la estructura de paquetes seguida tanto en el proyecto Android como en el de Eclipse.

Proyecto Eclipse

A continuación se muestra una descripción de los paquetes del proyecto de Eclipse y como se relacionan entre sí.

- Paquete *basedatossql*: Contiene las clases necesarias para crear y manejar la base de datos SQLite.
- Paquete *creador*: Contiene la clase que contiene el *main* del proyecto. Hace uso de los paquetes dbpedia y webscraping para extraer toda la información necesaria por la aplicación.
- Paquete *dbpedia*: Contiene las clases necesarias para extraer la información de la DBpedia. Hace uso del paquete traductor para traducir al inglés la información. Hace uso del paquete basededatosql para almacenar la información consultada en la base de datos.
- Paquete *traductor*: Contiene las clases necesarias para implementar un traductor automático.
- Paquete *webscraping*: Contiene las clases necesarias realizar la extracción de las claves dicotómicas mediante técnicas de *Web Scraping*. Hace uso del paquete traductor para traducir las claves dicotómicas.

En la figura C.12 se muestra el diagrama de paquetes del proyecto de Eclipse.

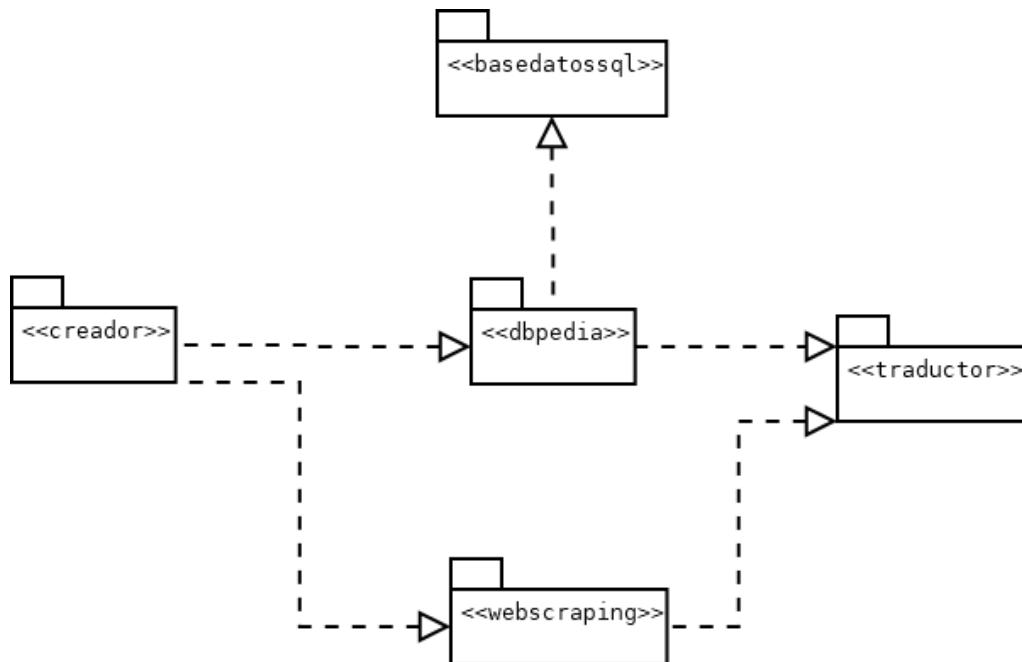


Figura C.12: Diagrama de paquetes del proyecto de Eclipse.

Proyecto Android

A continuación se muestra una descripción de los paquetes del proyecto Android y como se relacionan entre sí.

- Paquete *basedatos*: Paquete que contiene todas las clases necesarias para acceder a los datos de la aplicación.
- Paquete *clasificador*: Paquete que contiene las clases y actividades relacionadas con la implementación y uso del clasificador de imágenes.
- Paquete *clavedicotomica*: Paquete que contiene las clases relacionadas con mostrar las claves dicotómicas. Hace uso del paquete base de datos para extraer las claves dicotómicas del fichero serializado. Usa el paquete tarjetasclaves para implementar el listado de claves.
- Paquete *elegirclaves*: Paquete que contiene las clases relacionadas con el filtrado de géneros de la clave general.
- Paquete *informacion*: Paquete que contiene las clases relacionadas con mostrar la información de las distintas especies manejadas por la

aplicación. Hace uso del paquete basedatos para extraer la información de la base de datos SQLite y acceder a las imágenes de las setas, situadas en los directorios externos. Usa el paquete tarjetassetas para implementar el listado de especies de setas.

- Paquete *lanzador*: Paquete que recoje la clase lanzadora de la aplicación.
- Paquete *resultados*: Paquete que contiene las clases relacionadas con mostrar los resultados obtenidos por el clasificador. Hace uso del paquete basedatos para acceder a las imágenes de las setas, situadas en los directorios externos.
- Paquete *tarjetasclaves*: Paquete que contiene las clases relacionadas con mostrar el listado de claves dicotómicas disponibles en la aplicación.
- Paquete *tarjetassetas*: Paquete que contiene las clases relacionadas con mostrar el listado de especies disponibles en la aplicación.

En la figura C.13 se muestra el diagrama de paquetes del proyecto Android.

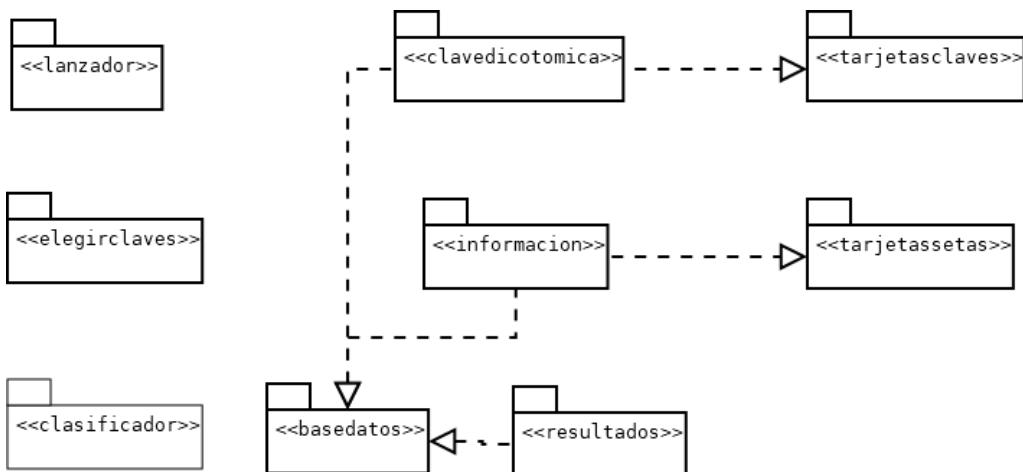


Figura C.13: Diagrama de paquetes del proyecto Android.

C.5. Diseño de la interfaz

En esta sección se van a mostrar los prototipos que se realizaron para diseñar la interfaz y un ejemplo de como es la interfaz final de la aplicación Android.

Prototipado

El prototipado de la aplicación se realizó de manera manual, dibujando un diseño inicial de lo que iba a ser la interfaz de cada actividad. En estas tarjetas se dibujo el diseño con y sin menú, así como la vista horizontal y vertical de la aplicación. A continuación se muestran las tarjetas de las diferentes actividades.

- Actividad Lanzadora C.14:

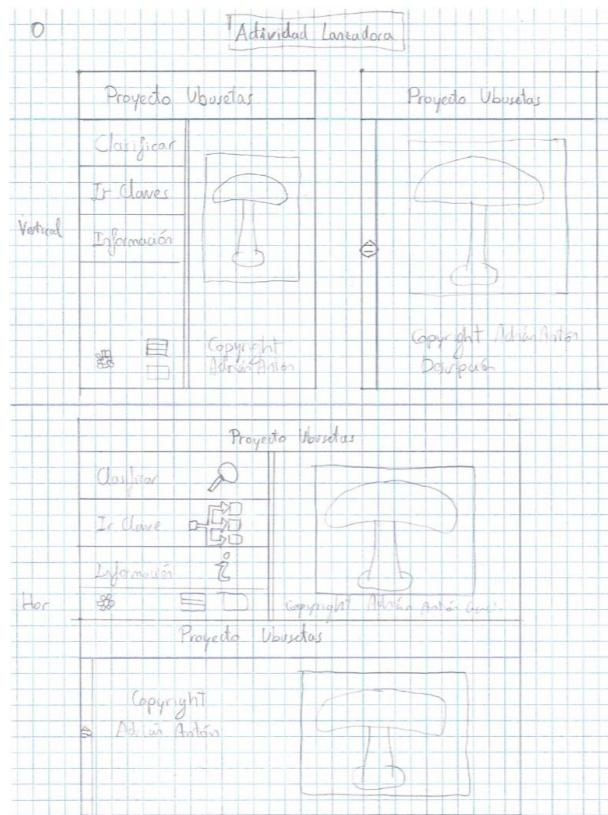


Figura C.14: Prototipo actividad Lanzadora.

■ Actividad Recoger Foto C.15:

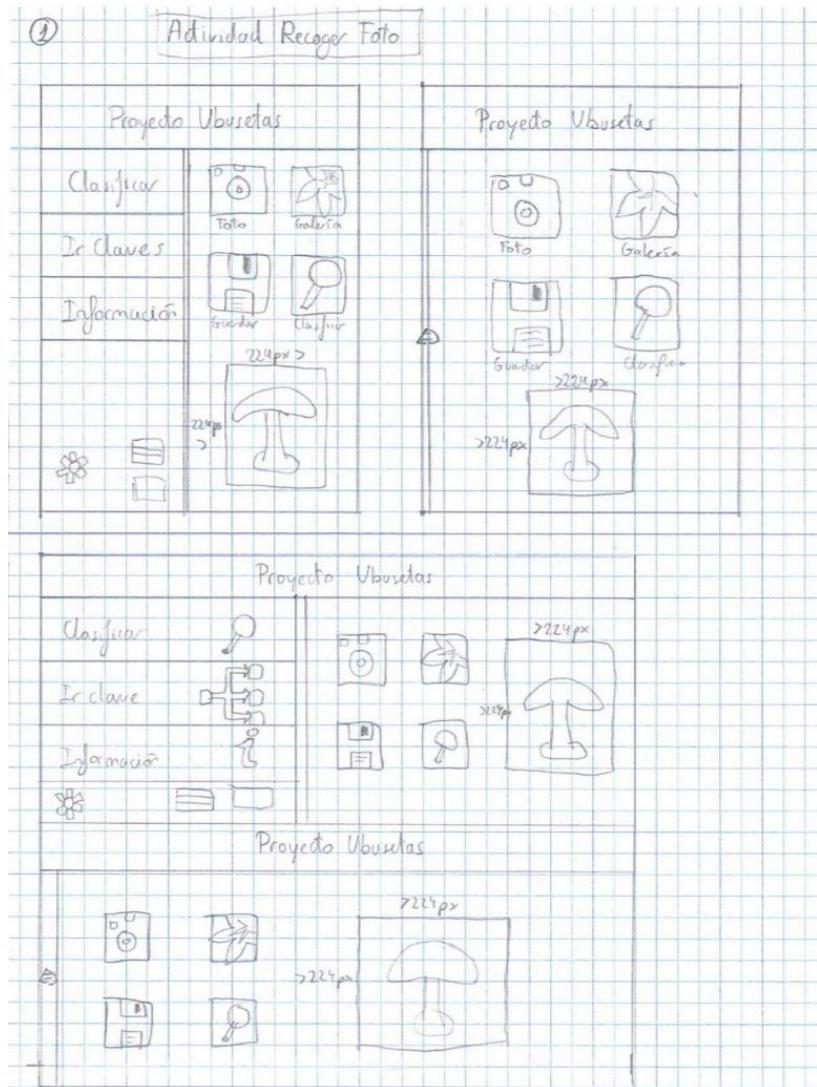


Figura C.15: Prototipo actividad Recoger Foto.

■ Actividad Mostrar Claves C.16:

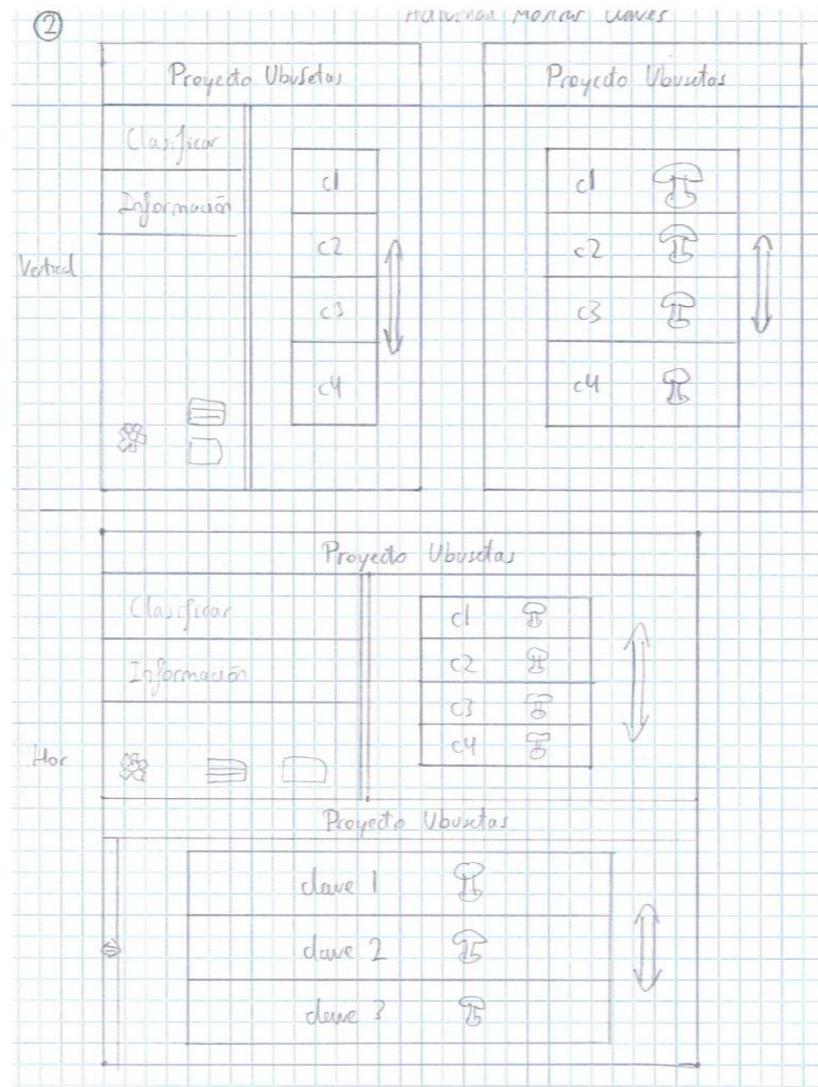


Figura C.16: Prototipo actividad Mostrar Claves.

■ Actividad Mostrar Setas C.17:

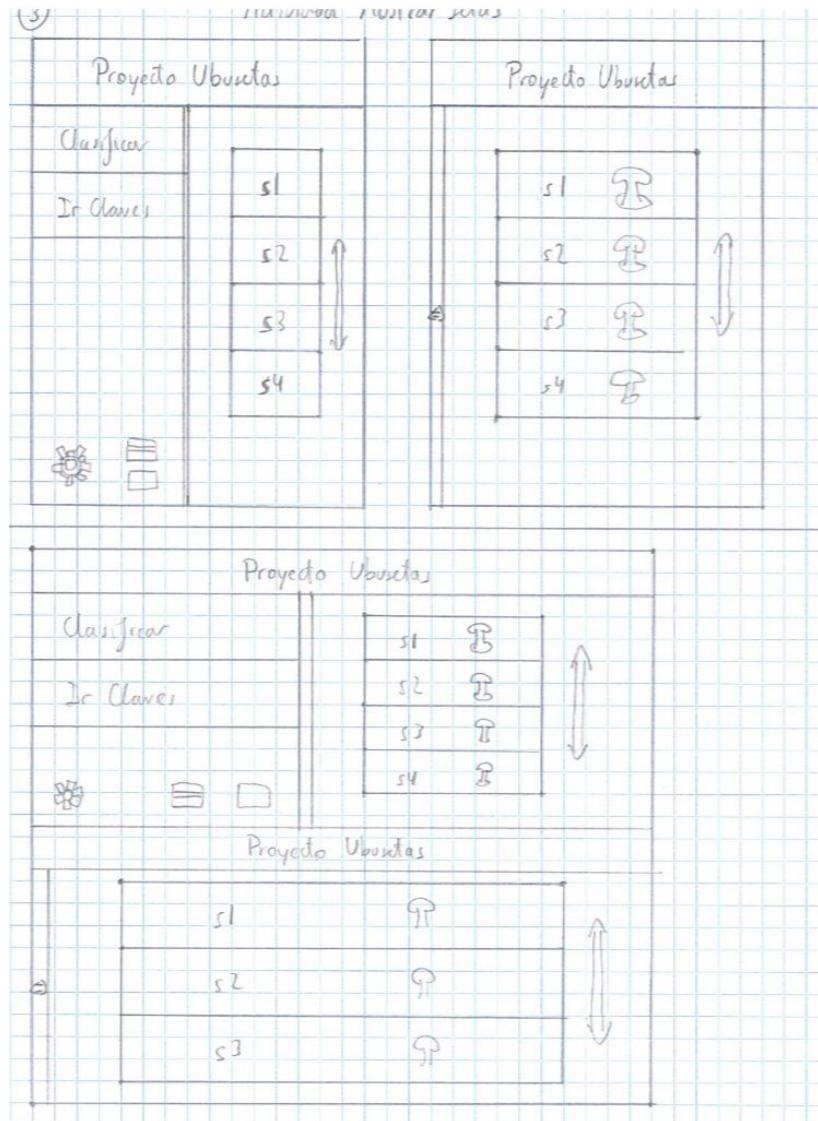


Figura C.17: Prototipo actividad Mostrar Setas.

■ Actividad Mostrar Resultados C.18:

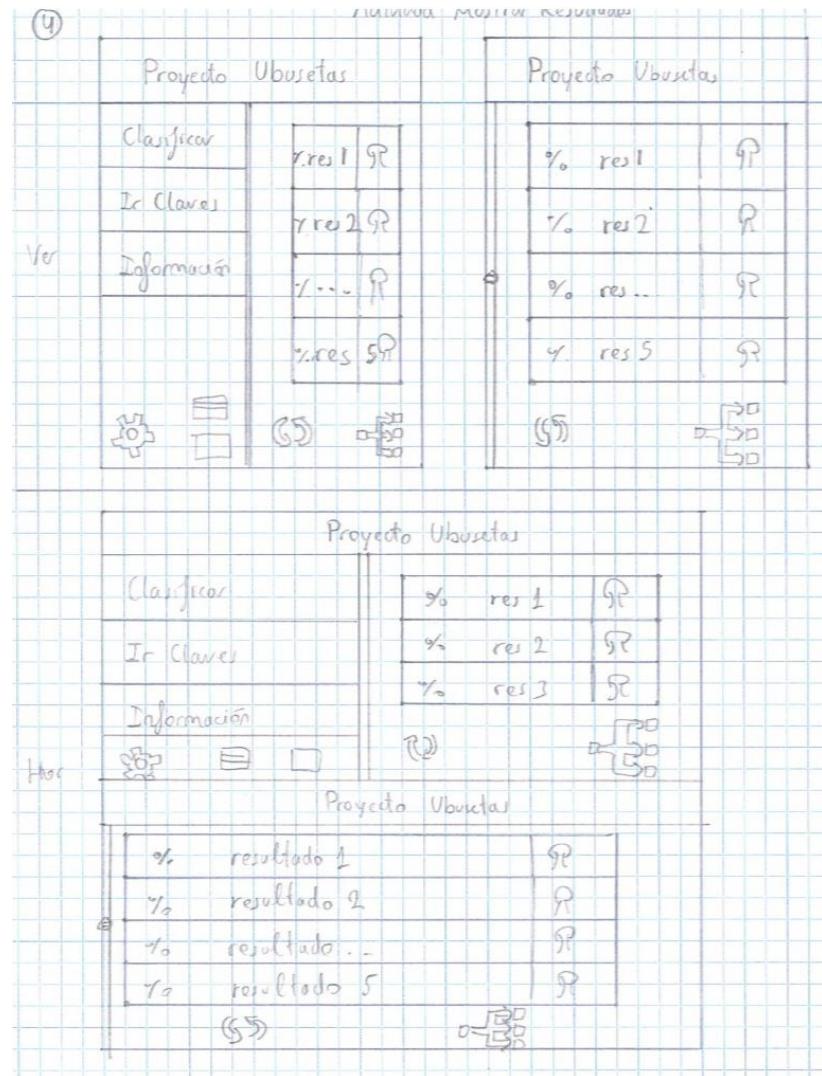


Figura C.18: Prototipo actividad Mostrar Resultados.

■ Actividad Clave Dicotómica C.19:

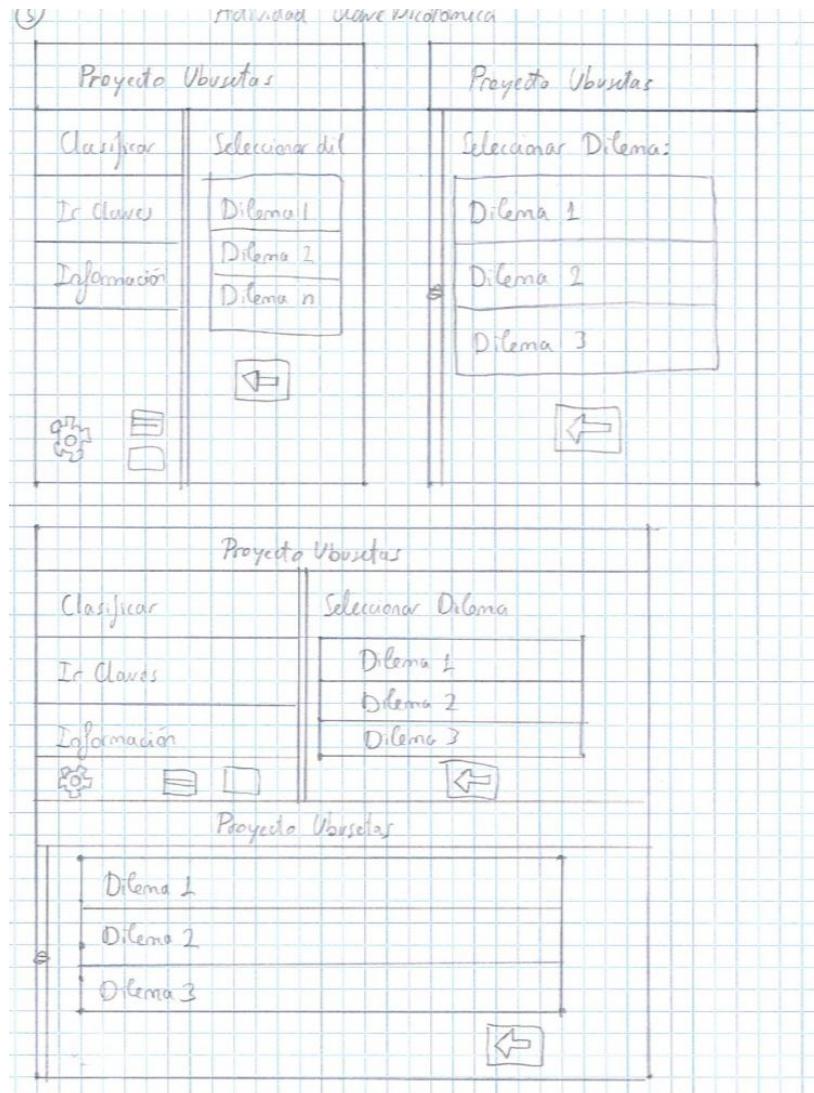


Figura C.19: Prototipo actividad Clave Dicotómica.

■ Actividad Mostrar Información C.20:

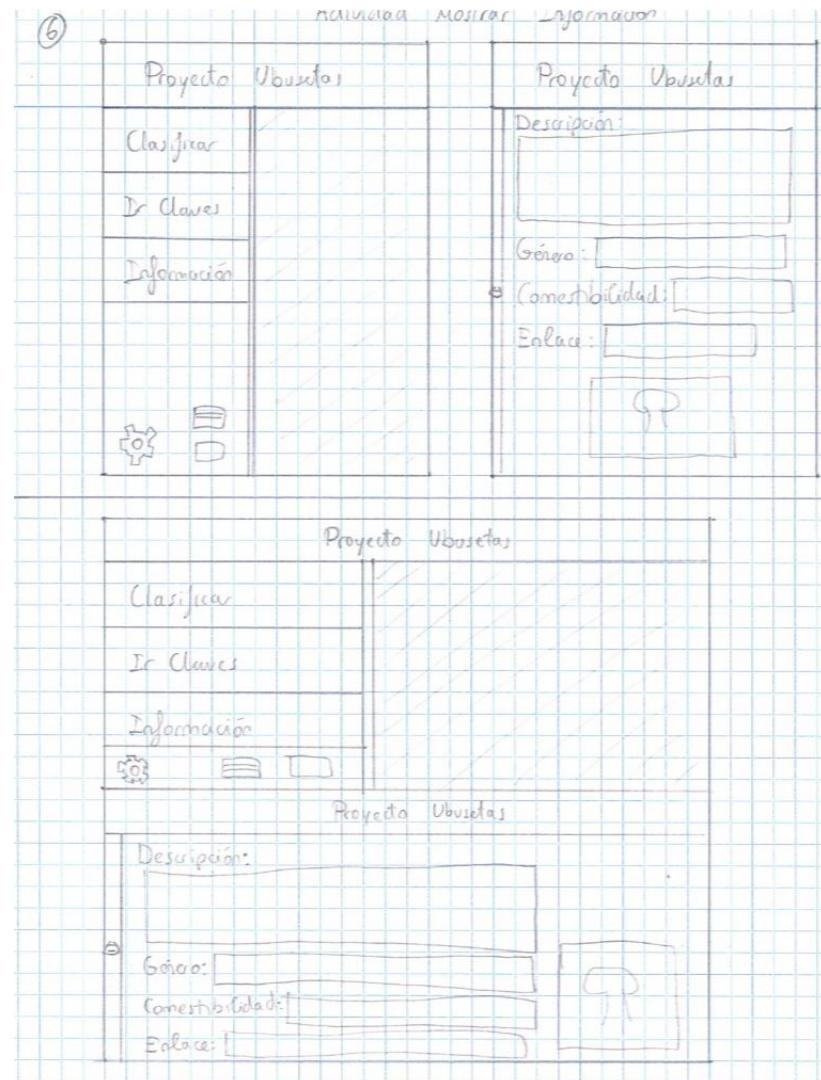


Figura C.20: Prototipo Mostrar Información.

- Actividad Elegir Claves C.21:

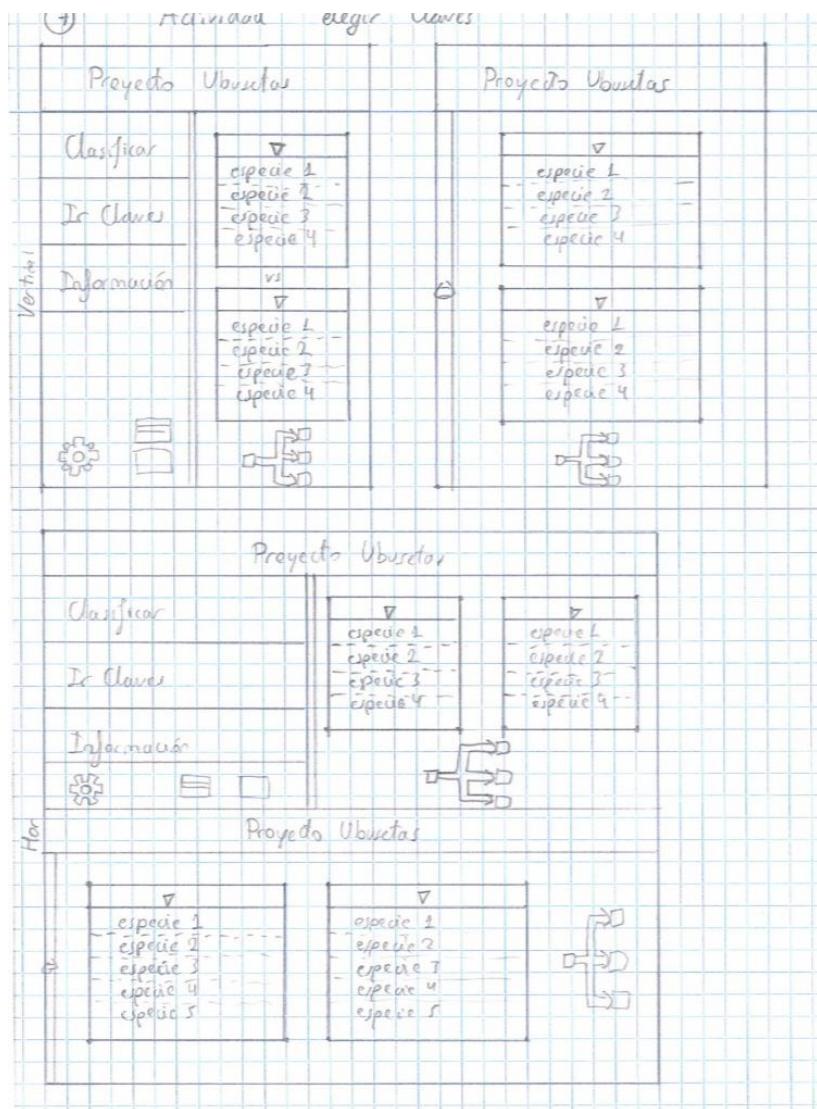


Figura C.21: Prototipo Elegir Claves.

■ Actividad Mostrar Comparativa C.22:

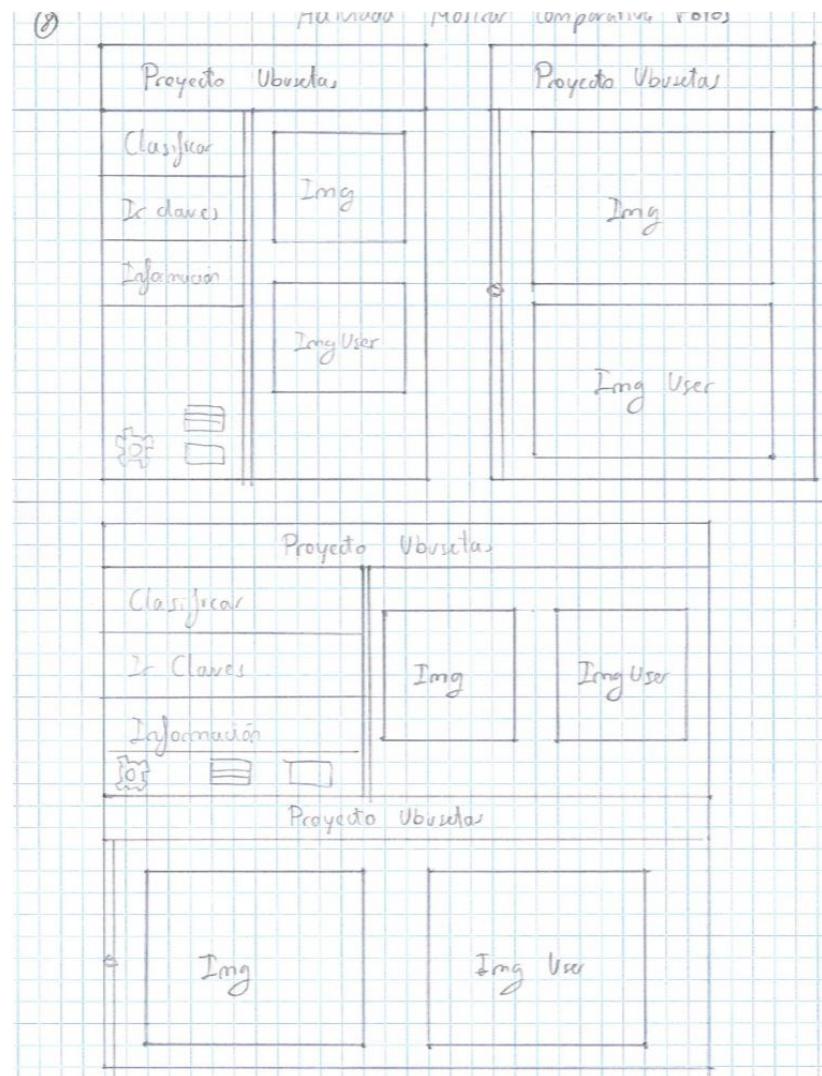


Figura C.22: Prototipo Mostrar Comparativa.

- Diagrama de flujo entre las actividades C.23:

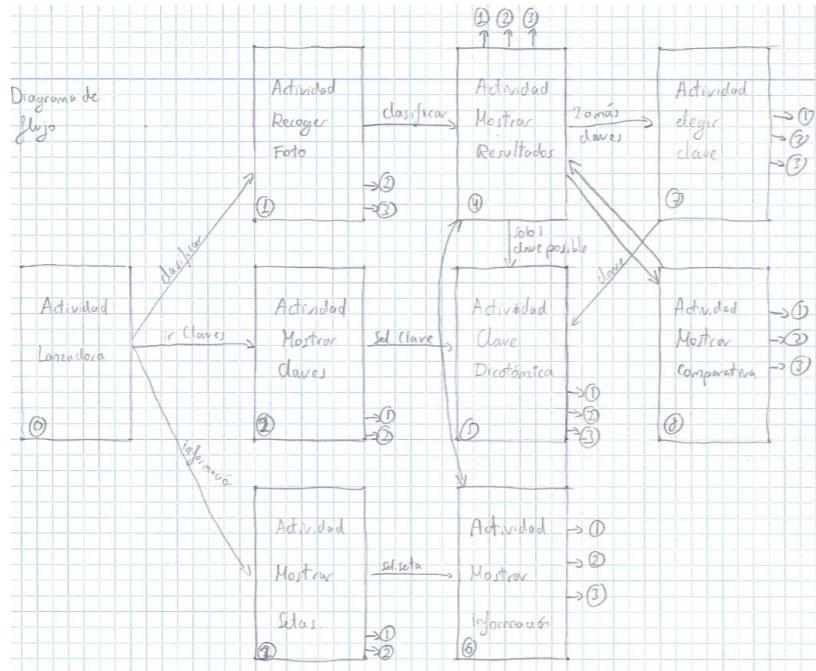


Figura C.23: Diagrama de flujo.

Diseño final

A continuación se muestra un ejemplo del diseño final de la actividad lanzadora en los siguientes estados:

- Sin el menú desplegado en vertical C.24.
- Con el menú desplegado en vertical C.25.
- Sin el menú desplegado en horizontal C.26.
- Con el menú desplegado en horizontal C.27.



Figura C.24: Vertical sin menú.

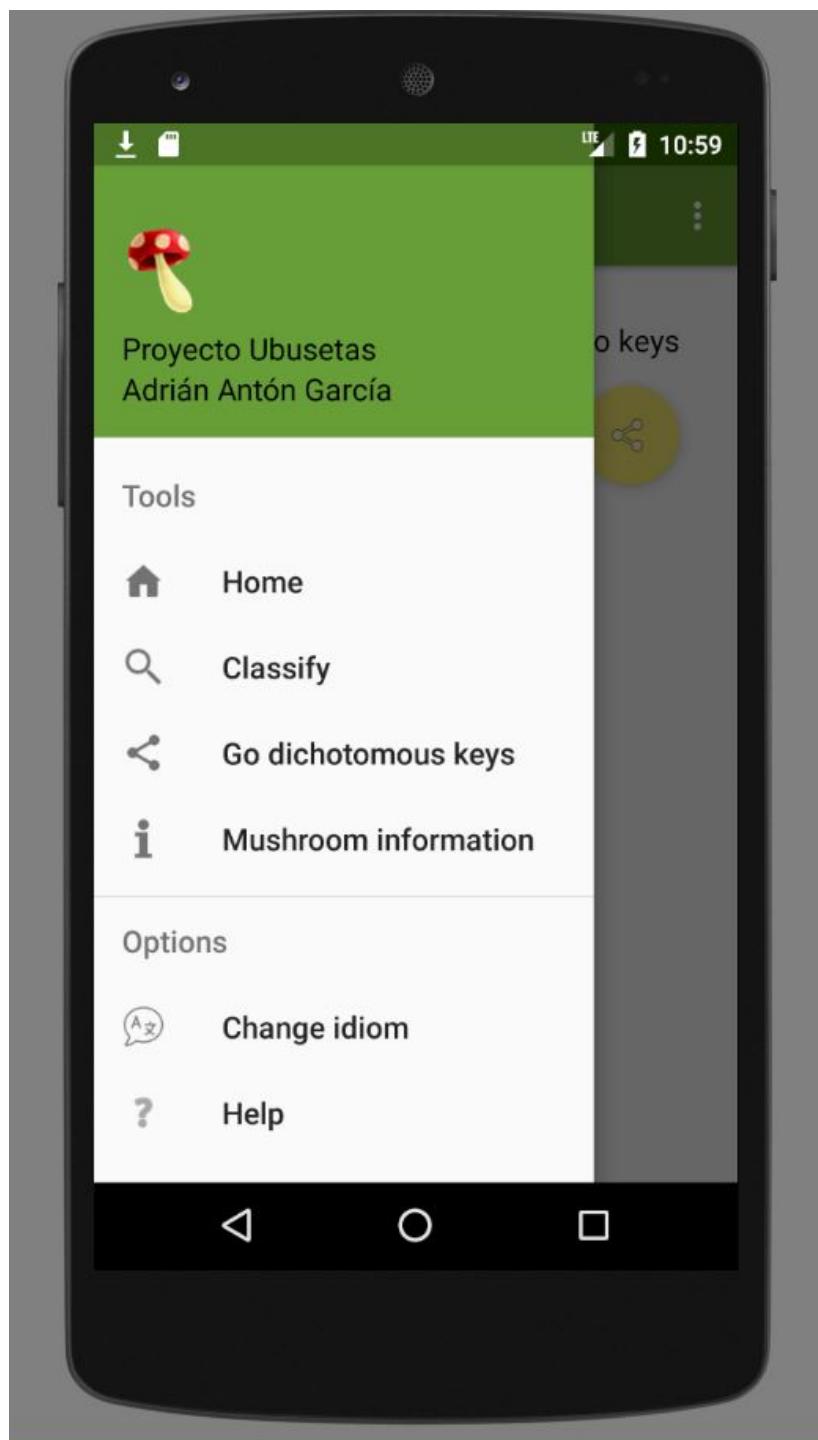


Figura C.25: Vertical con menú.



Figura C.26: Horizontal sin menú.

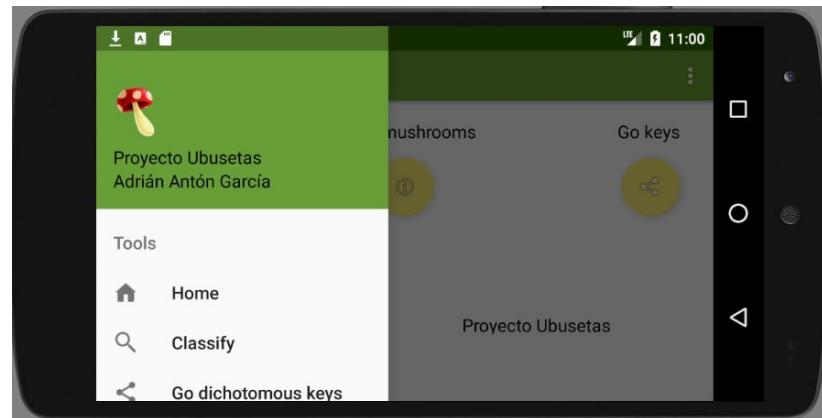


Figura C.27: Horizontal con menú.

Consideraciones interfaz

La interfaz se ha desarrollado para adaptarse a múltiples tamaños de pantalla, pero esta opción solo se ha podido probar con unos pocos modelos de móviles por lo que es probable que falle para alguna resolución en concreto.

La aplicación ha sido diseñada para funcionar tanto en modo apaisado como en vertical. Se han intentando seguir las directrices de *Material Design* en cuanto al diseño.

Apéndice D

Documentación técnica de programación

D.1. Introducción

Sección en la que se va a explicar la estructura de directorios que sigue la entrega del proyecto y a continuación, se desarrollarán los pasos necesarios para compilar y ejecutar los componentes del proyecto. El objetivo de esta sección es mostrar los pasos que se deben seguir para instalar todos los componentes necesarios para seguir desarrollando este proyecto.

D.2. Estructura de directorios

A continuación se van a listar los directorios que forman este proyecto y que están en el directorio raíz.

- **\Proyecto Eclipse:** Esta carpeta contiene los códigos fuente de la aplicación de Eclipse, el nombre del proyecto es *ExtraccionDatos*.
- **\Proyecto Android:** Esta carpeta contiene los códigos fuente de la aplicación Android, el nombre del proyecto es *ProyectoUbusetas*.
- **\Herramientas adicionales:** Contiene las librerías y proyectos externos *open source* usados en este proyecto.
 - **\Herramientas adicionales\image_retraining:** Esta carpeta contiene el proyecto de *Tensorflow* usado para entrenar los clasificadores de imágenes.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

- \Herramientas adicionales\apache-jena-3.4.0: Las librerías de Apache Jena.
 - \Herramientas adicionales\jaunt1.3.7: Las librerías de Apache Jaunt.
 - \Herramientas adicionales\json-20171018: Las librerías de Json.
 - \Herramientas adicionales\sqlite-jdbc-3.20.0: El driver de la base de datos SQLite.
- \Git: Contiene una copia del repositorio de Github del proyecto.
 - \Documentación: Contiene la documentación del proyecto, esta carpeta contiene los siguientes elementos:
 - \Documentación\memoria.pdf: La memoria del proyecto.
 - \Documentación\anexos.pdf: Los anexos del proyecto.
 - \Documentación\fuentes latex: Carpeta con los códigos de latex usados para generar la documentación.
 - \Documentación adicional: Contiene las siguientes dos carpetas con los *javadoc* de los proyectos.
 - \Documentación adicional\javadoc proyecto Eclipse: Esta carpeta contiene la documentación del proyecto Eclipse.
 - \Documentación adicional\javadoc proyecto Android: Esta carpeta contiene la documentación del proyecto Android.
 - \Binarios\aplicación Android: Carpeta donde se encuentra el instalador de la aplicación Android llamado *UBUsetas1.0.apk*
 - \Tutoriales: Contiene tutoriales para instalar ciertas herramientas y librerías necesarias para el funcionamiento del proyecto.

D.3. Manual del programador

Manual donde se va a explicar como realizar las tareas que llevaría a cabo el desarrollador o administrador de la aplicación.

Realización de las consultas a la DBpedia

En este apartado se va a explicar los pasos que hay que seguir para descargar la información de las especies de setas, a través del proyecto de Eclipse, y almacenarlas en una base de datos SQLite para su posterior exportación a la aplicación Android.

Para realizar esta tarea es necesario haber creado antes una base de datos *SQLite*. En la carpeta \Tutoriales se encuentra el pdf llamado "*Instalación SQLite windows*" en el que se explica como instalar y crear una base de datos *SQLite*.

Una vez tengamos la base de datos, deberemos especificar al programa la ruta donde se encuentra. Para ello editamos la clase *CreadorBD.java* indicándole en la siguiente línea D.1 la ruta.

```

26 public class CreadorBD {
27     // logger del creadorBD
28
29     private final static Logger logger = Logger.getLogger(CreadorBD.class);
30
31     // clase de la web semántica
32
33     private DBpedia dp = null;
34
35     // clase de la base de datos
36
37     private BDsql bd = null;
38
39     // urls para crear la base de datos
40
41     private String urlFichero = "src\\main\\java\\nombresSetas.txt";
42     private String urlBaseDatos = "jdbc:sqlite:C:\\sqlite\\DBsetas\\DBsetas.db";
43
44     // Nombres de las tablas
45

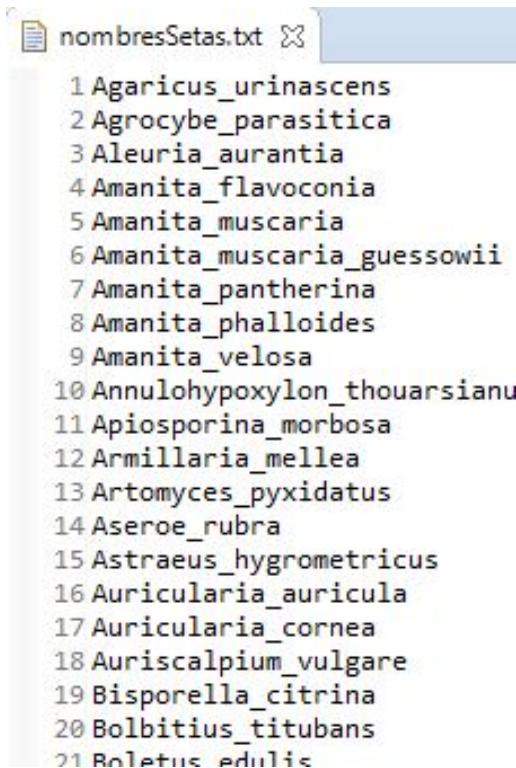
```

Figura D.1: Ruta de la base de datos.

Una vez hecho esto, deberemos ejecutar el *main* que encontramos en la clase *src/main/java/dbpedia/DBpedia.java*. El programa realizará las consultas a la *DBpedia* y guardará la información de las especies en la base de datos *SQLite* especificada. Este proceso tardará unos minutos, dependiendo de la velocidad de la conexión a Internet.

La lista de especies de setas a consultar se encuentra en el fichero que encontramos en la ruta *src/main/java/nombresSetas.txt* del proyecto. Encontraremos la siguiente estructura D.2 en la que podremos añadir o quitar especies para consultar a la DBpedia.

SOPÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN



The screenshot shows a text file titled "nombresSetas.txt" containing a list of 21 mushroom species names, each preceded by a number from 1 to 21. The species listed are: Agaricus_urinascens, Agrocybe_parasitica, Aleuria_aurantia, Amanita_flavoconia, Amanita_muscaria, Amanita_muscaria_guessowii, Amanita_pantherina, Amanita_phalloides, Amanita_velosa, Annulohypoxylon_thouarsianum, Apiosporina_morbosa, Armillaria_mellea, Artomyces_pygidatus, Aseroe_rubra, Astraeus_hygrometricus, Auricularia_auricula, Auricularia_cornea, Auriscalpium_vulgare, Bisporaella_citrina, Bolbitius_titubans, and Boletus_edulis.

```
1 Agaricus_urinascens
2 Agrocybe_parasitica
3 Aleuria_aurantia
4 Amanita_flavoconia
5 Amanita_muscaria
6 Amanita_muscaria_guessowii
7 Amanita_pantherina
8 Amanita_phalloides
9 Amanita_velosa
10 Annulohypoxylon_thouarsianum
11 Apiosporina_morbosa
12 Armillaria_mellea
13 Artomyces_pygidatus
14 Aseroe_rubra
15 Astraeus_hygrometricus
16 Auricularia_auricula
17 Auricularia_cornea
18 Auriscalpium_vulgare
19 Bisporaella_citrina
20 Bolbitius_titubans
21 Boletus_edulis
```

Figura D.2: Listado de especies de setas.

Por último, para instalar la base de datos en la aplicación Android, deberemos copiar nuestro fichero *DBsetas.db* (o el nombre que le hayamos dado) dentro de la carpeta */app/src/main/assets/databases* del proyecto Android.

Extracción de las claves dicotómicas

En esta sección se explicará como ejecutar el proyecto Eclipse para que extraiga las claves dicotómicas, mediante *web scraping*, de la página web <http://www.avelinosetas.info/claves.php>.

Para realizar esta tarea, simplemente deberemos ejecutar el main que se encuentra en la clase *src/main/java/webscraping/ClaveDicotomica.java* del proyecto de Eclipse.

Una vez hayamos ejecutado este programa, se nos habrán creado dos archivos en la raíz del proyecto:

- claves.dat: contiene las claves dicotómicas en Español.
- clavesEn.dat: contiene las claves dicotómicas en Inglés.

Por último, para instalar las claves en la aplicación Android, deberemos copiar nuestros ficheros *claves.dat* y *clavesEn.dat* dentro de la carpeta */app/src/main/assets/claves* del proyecto Android.

Entrenamiento del clasificador

En este apartado se explicará como usar el proyecto de Tensorflow para entrenar nuestro propio clasificador de imágenes. Este proyecto lo podremos encontrar en la carpeta *\Herramientas adicionales\image_retraining*. Tiene la siguiente estructura D.3.

an anton garcia > Dropbox > Universidad5 > Proyecto Ubu > Entrega > Herramientas adicionales > image_retraining			
Nombre	Fecha de modifica...	Tipo	Tamaño
comandos	08/01/2018 16:46	Carpeta de archivos	
data	08/01/2018 16:46	Carpeta de archivos	
images	08/10/2017 20:58	Carpeta de archivos	
init.py	27/07/2017 18:25	Python File	0 KB
BUILD	27/07/2017 18:25	Archivo	2 KB
label_image.py	27/07/2017 18:25	Python File	5 KB
retrain.py	27/07/2017 18:25	Python File	54 KB
retrain_test.py	27/07/2017 18:25	Python File	6 KB

Figura D.3: Estructura Image Retraining.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

Para hacer uso de este proyecto necesitaremos tener instalados Python y Tensorflow¹ en nuestro sistema.

Lo primero que deberemos hacer es colocar en la carpeta *images* del proyecto nuestras imágenes divididas en carpetas. En cada carpeta colocaremos las imágenes que se correspondan con una misma especie de seta y llamaremos a esa carpeta con el nombre de la especie **D.2**.

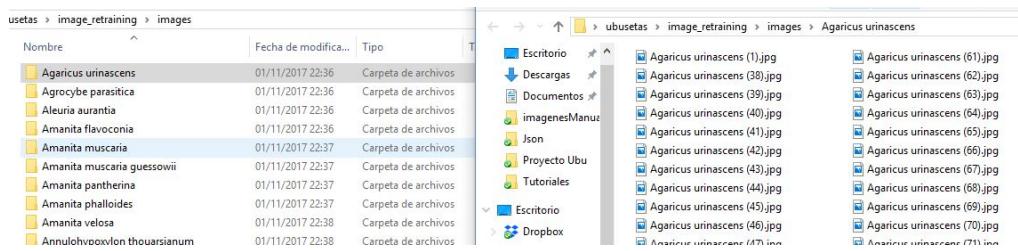


Figura D.4: Estructura Image Retraining.

Si queremos hacer uso del *data augmentation*, deberemos modificar los parámetros dentro del fichero *retrain.py*, según se indica en las descripciones de estos dentro del código.

Para entrenar el modelo deberemos abrir una consola y colocarnos en la carpeta *image_retraining*. Una vez aquí deberemos activar Tensorflow con el siguiente comando **D.5**.

```
C:\Users\adrit\Desktop\ubusetas\image_retraining>activate Tensorflow  
(Tensorflow) C:\Users\adrit\Desktop\ubusetas\image_retraining>
```

Figura D.5: Activación Tensorflow.

Cuando hayamos activado *Tensorflow* podremos ejecutar el fichero *retrain.py* para entrenar el modelo. Deberemos especificarle el modelo que queremos entrenar, por ejemplo, para entrenar el *Mobilenet-224* ejecutaríamos el siguiente comando **D.6**.

¹<https://www.tensorflow.org/install/>

D.4. COMPILACIÓN, INSTALACIÓN Y EJECUCIÓN DEL PROYECTO⁸³

```
(Tensorflow) C:\Users\adrit\Desktop\ubusetas\image_retraining>python retrain.py --architecture mobilenet_1.0_224 --image_dir images
INFO:tensorflow:Looking for images in 'Agaricus urinascens'
INFO:tensorflow:Looking for images in 'Agrocybe parasitica'
INFO:tensorflow:Looking for images in 'Aleuria aurantia'
INFO:tensorflow:Looking for images in 'Amanita flavoconia'
INFO:tensorflow:Looking for images in 'Amanita muscaria'
INFO:tensorflow:Looking for images in 'Amanita muscaria guessowii'
INFO:tensorflow:Looking for images in 'Amanita pantherina'
INFO:tensorflow:Looking for images in 'Amanita phalloides'
INFO:tensorflow:Looking for images in 'Amanita velosa'
INFO:tensorflow:Looking for images in 'Annulohypoxylon thouarsianum'
INFO:tensorflow:Looking for images in 'Apiosporina morbosa'
INFO:tensorflow:Looking for images in 'Armillaria mellea'
INFO:tensorflow:Looking for images in 'Asterophora lycoperdon'
```

Figura D.6: Entrenamiento Mobilenet.

Cuando el proceso acabe (puede tardar horas o días según la configuración), se nos generarán los siguientes dos archivos en la carpeta C:\tmp:

- output_graph.pb: El modelo entrenado.
- output_labels.txt: Los *labels* del modelo.

Para instalar nuestro modelo Mobilenet en la aplicación Android debaremos colocar estos dos archivos en la carpeta */app/src/main/assets/mobilenet-224* del proyecto Android.

D.4. Compilación, instalación y ejecución del proyecto

En esta sección se desarrollarán los pasos necesarios para instalar tanto el proyecto Eclipse como el proyecto Android.

Proyecto Eclipse

En este apartado se explicará como instalar el proyecto de Eclipse llamado *ExtraccionDatos*. Este proyecto lo podemos encontrar en la carpeta *Proyecto Eclipse*, deberemos extraerlo para poder importarlo.

En el proyecto se ha usado la versión de Eclipse Neon.3 Release (4.6.3). Necesitaremos una versión de Eclipse² igual o superior a esta para instalarlo. Además necesitaremos tener instalado Maven en Eclipse³.

Para instalar el proyecto *ExtraccionDatos* deberemos importarlo como un proyecto Maven de la siguiente forma [D.7](#) [D.8](#).

²<http://www.eclipse.org/downloads/eclipse-packages/>

³<https://maven.apache.org/install.html>

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

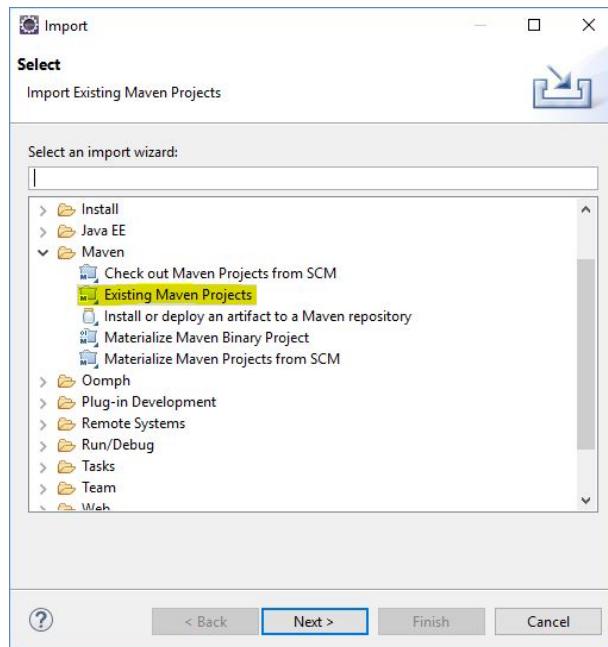


Figura D.7: Importación Eclipse.

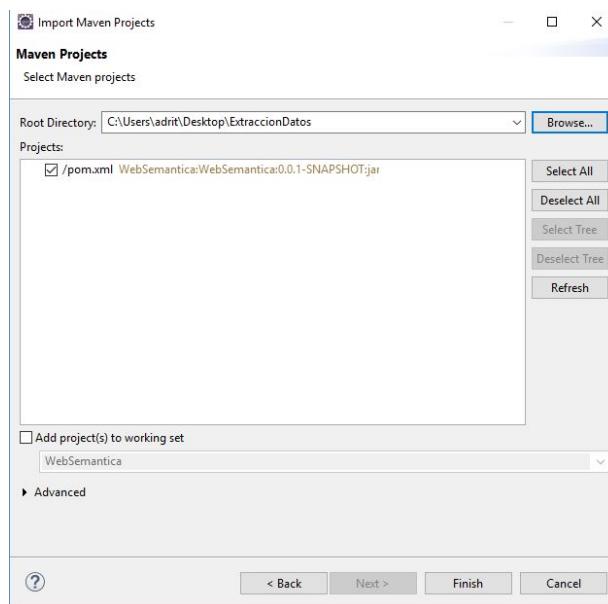


Figura D.8: Importación Eclipse.

D.4. COMPILACIÓN, INSTALACIÓN Y EJECUCIÓN DEL PROYECTO

Una vez importado el proyecto base deberemos importar las siguientes librerías al proyecto. Se ha creado un tutorial para cada una que muestra como se descargan e importan al proyecto.

- apache-jena-3.4.0: Es la librería que nos proporciona los métodos necesarios para realizar las consultas a la DBpedia. El tutorial se encuentra en el archivo *Tutoriales/Instalación librería Jena en Eclipse.pdf*.
- jaunt1.3.7: Esta librería nos permite hacer *web scrapings*sobre una página Web. Tiene la peculiaridad de que hay que actualizarla cada mes. El tutorial se encuentra en el archivo *Tutoriales/Instalación librería Jaunt en Eclipse.pdf*.
- json-20171018: Nos permite usar JSON en el proyecto. El tutorial se encuentra en el archivo *Tutoriales/Instalación librería Json.pdf*.
- sqlite-jdbc-3.20.0: Nos proporciona los métodos para acceder a la base de datos SQuite. El tutorial se encuentra en el archivo *Tutoriales/Instalación SQuite windows.pdf*.

Una vez realizadas las importaciones, ya podremos compilar el programa a través de Eclipse.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

Proyecto Android

En este apartado se explicará como se importa el proyecto Android llamado *ProyectoUbusetas*. Este proyecto lo podemos encontrar en la carpeta *Proyecto Android*, deberemos extraerlo para poder importarlo.

Necesitaremos tener instalado en el sistema el entorno de desarrollo Android Studio⁴. En este proyecto se ha usado la versión 3.0.1 de Android Studio.

Para importar el proyecto solo habrá que abrir con Android Studio el proyecto proporcionado, automáticamente se descargarán las dependencias necesarias y cuando este proceso acabe, el proyecto estará listo para compilarse.

Javadoc

Tanto el proyecto Android como el Eclipse tienen generados sus correspondientes ficheros Javadoc en los que se ha explicado el funcionamiento de todas las clases y métodos implementados.

D.5. Pruebas del sistema

Las pruebas implementadas se han desarrollado sobre el proyecto Android. Estas pruebas se han dividido en los siguientes 3 tipos:

- Pruebas Unitarias D.9: Se han creado pruebas unitarias para probar el correcto funcionamiento de los métodos de las actividades de manera individual. En principio estas pruebas solo cubren el 40 % D.10 del código debido a que no puedo probar la mayoría de los elementos de la interfaz con estas pruebas. Estas pruebas se han concentrado en probar los elementos más críticos y que no tienen que ver con la interfaz. Estos códigos no accesibles desde las pruebas unitarias se han probado con las pruebas de integración.
- Pruebas de integración D.11: Pruebas que se han creado para comprobar la correcta interacción entre las actividades y el correcto funcionamiento de la interfaz de la aplicación.
- Pruebas de rendimiento: Para las pruebas de rendimiento se uso la herramienta *Monkeyrunner*. Esta herramienta se configuró para que

⁴<https://developer.android.com/studio/index.html?hl=es-419>

ejecutarán 50.000 eventos aleatorios sobre la aplicación y comprobar si se bloqueaba en algún punto. La aplicación consiguió aguantar todos los eventos.

La descripción de cada prueba se encuentra en el javadoc del proyecto Android.

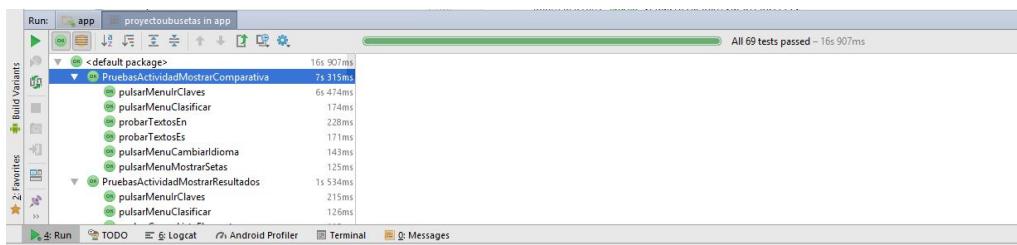


Figura D.9: PruebasUnitarias.

40% classes, 42% lines covered in package 'proyectoubusetas'			
Element	Class, %	Method, %	Line, %
basedatos	100% (3/3)	93% (15/16)	84% (93/110)
clasificador	60% (3/5)	30% (9/30)	28% (92/319)
clavedicotonica	100% (2/2)	55% (11/20)	56% (208/368)
elegirclaves	60% (3/5)	30% (7/23)	46% (91/194)
informacion	100% (2/2)	43% (7/16)	57% (146/255)
lanzador	100% (1/1)	50% (4/8)	64% (62/96)
resultados	33% (5/15)	13% (18/137)	25% (217/835)
tarjetasclaves	50% (2/4)	42% (6/14)	27% (13/48)
tarjetassetas	50% (2/4)	50% (8/16)	29% (16/55)
BuildConfig	0% (0/1)	0% (0/1)	0% (0/2)
R	5% (1/17)	100% (1/1)	77% (58/75)

Figura D.10: *Coverage* pruebas unitarias.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN



Figura D.11: Ejemplo prueba integración.

Apéndice E

Documentación de usuario

E.1. Introducción

En esta sección se explicará todo lo necesario para que los usuarios sean capaces de instalar y usar la aplicación de manera correcta.

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Bibliografía

- [1] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.
- [2] Wikipedia. Latex — wikipedia, la enciclopedia libre, 2015. [Internet; descargado 30-septiembre-2015].