



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**Visor de espectros
Documentación Técnica**



Presentado por Iván Iglesias Cuesta
en Universidad de Burgos — 27 de junio
de 2018

Tutor: Dr. José Francisco Díez Pastor
Cotutor: Dr. César Ignacio García Osorio

Índice general

Índice general	I
Índice de figuras	III
Índice de tablas	IV
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	1
A.3. Estudio de viabilidad	10
Apéndice B Especificación de Requisitos	11
B.1. Introducción	11
B.2. Objetivos generales	11
B.3. Catalogo de requisitos	12
B.4. Especificación de requisitos	13
Apéndice C Especificación de diseño	23
C.1. Introducción	23
C.2. Diseño de datos	23
C.3. Diseño procedimental	23
C.4. Diseño arquitectónico	23
Apéndice D Documentación técnica de programación	25
D.1. Introducción	25
D.2. Estructura de directorios	25
D.3. Manual del programador	25

D.4. Compilación, instalación y ejecución del proyecto	26
D.5. Pruebas del sistema	26
Apéndice E Documentación de usuario	27
E.1. Introducción	27
E.2. Requisitos de usuarios	27
E.3. Instalación	27
E.4. Manual del usuario	27
Bibliografía	29

Índice de figuras

A.1. Burndown del <i>sprint</i> 3	3
A.2. Burndown del <i>sprint</i> 4	4
A.3. Burndown del <i>sprint</i> 5	5
A.4. Burndown del <i>sprint</i> 6	6
A.5. Burndown del <i>sprint</i> 7	7
A.6. Burndown del <i>sprint</i> 8	8
A.7. Burndown del <i>sprint</i> 9	8
A.8. Burndown del <i>sprint</i> 10	9
A.9. Burndown del <i>sprint</i> 11	10
B.1. Diagrama de casos de uso	14

Índice de tablas

B.1. Iniciar sesión.	15
B.2. Cerrar sesión.	16
B.3. Subir dataset.	17
B.4. Subir espectro.	18
B.5. Visualizar dataset.	19
B.6. Visualizar espectro.	20
B.7. Guardar clasificador.	21
B.8. Predecir espectro.	22

Apéndice A

Plan de Proyecto Software

A.1. Introducción

Para que un proyecto se desarrolle con normalidad y con el menor número de imprevistos posibles es esencial que cuente con una fase de planificación. Aquí se estima el tiempo, trabajo y dinero que puede llegar a usarse durante la realización del proyecto. Para ello, se debe analizar en detalle cada parte del proyecto. De cara al futuro, el análisis del proyecto puede servir para predecir como de bien puede desarrollarse una continuación del mismo.

La planificación del proyecto consta de dos partes:

- **Planificación temporal:** en esta parte se analiza y planifica el tiempo que se va a dedicar a cada parte del proyecto, fecha de inicio y final aproximado, teniendo en cuenta el trabajo necesario para cada parte.
- **Estudio de viabilidad:** en esta parte se analiza como de viable es la realización del proyecto, se divide a su vez en dos apartados:
 - **Económica:** en esta parte se estiman los costes y los beneficios que puede suponer el proyecto.
 - **Legal:** en esta parte se analizan los conceptos legales del proyecto, como podrían ser las licencias del proyecto o la política de protección de datos.

A.2. Planificación temporal

La planificación temporal se organiza mediante *sprints*. Cada *sprint* dura una o dos semanas. Al terminar cada *sprint* se realiza una reunión con los

tutores para dar por terminado

Sprint 0

En este *sprint* marcó el inicio del proyecto. La lista de tareas está disponible en [Sprint 0](#)¹. En reuniones previas se habló con los tutores en que iba a consistir en proyecto, pero no estaba claro con que tecnologías desarrollarlo. Se decidió hacer una evaluación de las tecnologías posibles y crear unos prototipos básicos. Todas las tareas se completaron a tiempo.

Sprint 1

En este *sprint* se habló sobre el despliegue de la aplicación. Los tutores comentaron que en proyectos webs anteriores el despliegue se solía dejar para las etapas finales del proyecto, haciendo que todos los problemas asociados surjan en esas finales, retrasando el despliegue y, a veces, no llegar a desplegar la aplicación.

Se conocía la plataforma Heroku para ello y se probó a usarla, adicionalmente se buscaron otras alternativas. También se usó el prototipo escogido para crear el proyecto definitivo y se trabajó en la memoria. La última parte fue estudiar las guías de estilo de Python, aplicarlas en los prototipos y documentar su código.

La lista de tareas está disponible en [Sprint 1](#)². Todas las tareas se completaron a tiempo.

Sprint 2

Dado que en la aplicación se necesita que los usuarios suban contenido, se necesita control de usuarios, en este *sprint* se investigaron formas de ofrecerlo. También debido a la necesidad de disponer almacenamiento persistente se tuvo que mirar otras formas de despliegue y buscar como He-

¹<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/1?closed=1>

²<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/2?closed=1>

roku lo ofrece, que de serie no lo hace. Al final se decidió cambiar a Nanobox.

La lista de tareas está disponible en [Sprint 2](#)³. Todas las tareas se completaron a tiempo.

Sprint 3

En este *sprint* se podría decir que comienza el desarrollo del proyecto, basado en el prototipo. Como tal se mejoró el aspecto visual de la aplicación, se cambió la estructura para tener partes diferenciadas y mantenibles, añadir control de usuarios y mejorar la subida de ficheros.

También se planteó añadir subir y visualizar un *dataset* completo, escribir el manual de despliegue y los casos de uso.

La lista de tareas está disponible en [Sprint 3](#)⁴. El manual de despliegue y la subida de *datasets* no se pudieron completar. El gráfico *burndown* del *sprint* se ve en la figura A.1.

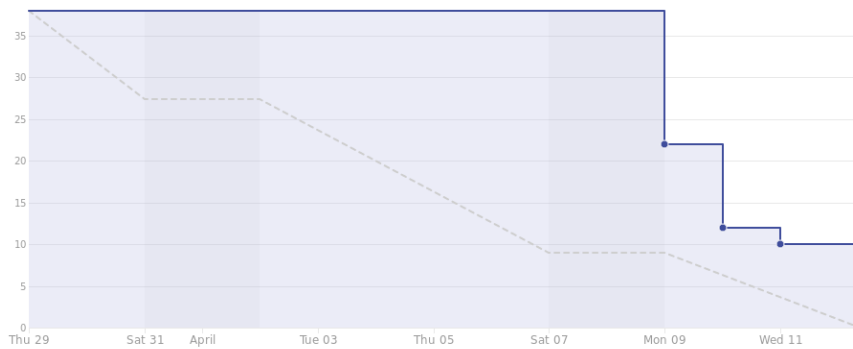


Figura A.1: Burndown del *sprint* 3

³<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/3?closed=1>

⁴<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/4?closed=1>

Sprint 4

En este *sprint* se completan las tareas que no habían dado tiempo del *sprint* anterior y se planteó usar una base de datos en lugar de almacenamiento para guardar los datos, por lo que se realizó una comparación entre formas de almacenar los datos. También se añadió la opción de borrar un *dataset* ya almacenado.

Al igual que en el *sprint* anterior se estructuró la aplicación como conjunto, en este *sprint* se estructura la parte de visualización. También se arreglaron dos *bugs* que se introdujeron en el *sprint* anterior en la aplicación desplegada.

La lista de tareas está disponible en [Sprint 4⁵](#). Todas las tareas se completaron a tiempo. El gráfico *burndown* del *sprint* se ve en la figura A.2.



Figura A.2: Burndown del *sprint* 4

Sprint 5

De la comparación del *sprint* anterior se decidió usar MongoDB para el almacenamiento, por lo cual la mayoría de los esfuerzos se centraron en adaptar la aplicación para usar MongoDB en todos sus aspectos: guardar, borrar y coger los datos. También se solucionó un *bug* en la parte de visualización.

⁵<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/5?closed=1>

La lista de tareas está disponible en [Sprint 5](#)⁶. Todas las tareas se completaron a tiempo. El gráfico *burndown* del *sprint* se ve en la figura A.3.

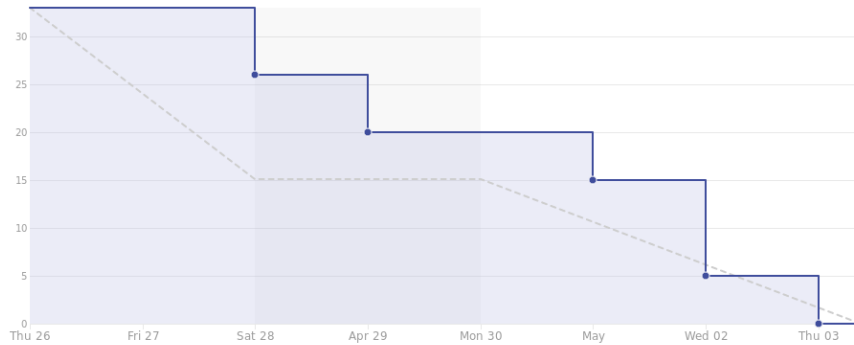


Figura A.3: Burndown del *sprint* 5

Sprint 6

Para este *sprint* se planteó cambiar la forma en la que se guardan los *datasets*, de forma que cuando el sistema de aprendizaje automático esté implementado sea más sencillo pasar los datos para el entrenamiento. También se añadió soporte de comentarios en el *dataset*. Debido a un cambio en como la geóloga organizaba sus datos, se tuvo de adaptar la subida de *datasets* a este cambio. Por último se empezó a añadir controles en la parte de visualización para el procesamiento de los espectros.

La lista de tareas está disponible en [Sprint 6](#)⁷. La tarea de los controles no se completó a tiempo. El gráfico *burndown* del *sprint* se ve en la figura A.4.

⁶<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/6?closed=1>

⁷<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/7?closed=1>

Figura A.4: Burndown del *sprint* 6

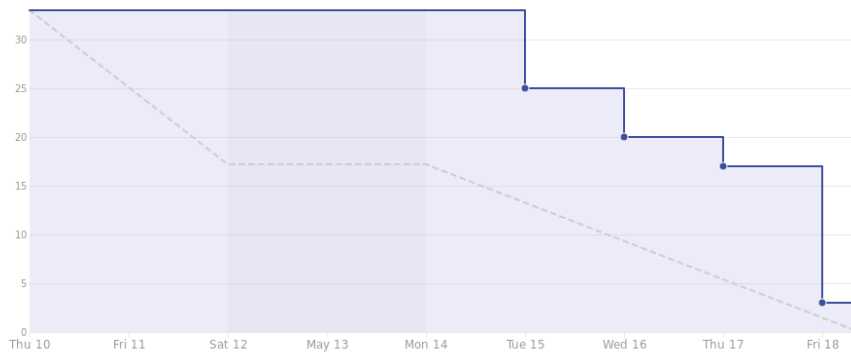
Sprint 7

Este *sprint* se centró en completar la tarea del *sprint* anterior, cambiar la parte de visualización para que se muestre en un tabla los ejemplos subidos y sus metadatos, documentar el código y avanzar en la memoria y anexos.

Durante el *sprint*, se vio que la tarea sobre procesamiento de datos era demasiado extensa, dividiéndola en dos partes, la primera que sería añadir los controles para el procesamiento en la interfaz, y una segunda para añadir el código de procesamiento de datos, para realizar en el siguiente *sprint*.

La lista de tareas está disponible en [Sprint 7⁸](https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/8?closed=1). No dio tiempo a escribir la introducción de la memoria. El gráfico *burndown* del *sprint* se ve en la figura A.5.

⁸<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/8?closed=1>

Figura A.5: Burndown del *sprint* 7

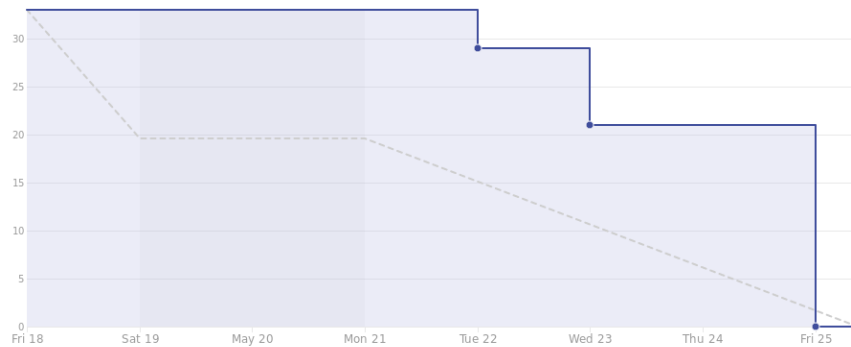
Sprint 8

Este *sprint* se centró en completar la tarea del *sprint* anterior, arreglar un *bug* del despliegue, añadir al proyecto el código de procesamiento de datos y enlazarlo a los controles, añadir instrucciones de esta parte y ampliar la planificación temporal con links al repositorio y capturas de los gráficos *burndown*.

Se recuerda que el código de procesamiento es gran parte de los resultados del proyecto previo en las colaboraciones.

La lista de tareas está disponible en [Sprint 8⁹](https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/9?closed=1). Todas las tareas se completaron a tiempo. El gráfico *burndown* del *sprint* se ve en la figura A.6.

⁹<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/9?closed=1>

Figura A.6: Burndown del *sprint* 8

Sprint 9

Este *sprint* se centró en añadir la funcionalidad de los espectros individuales: subida, visualización, procesado y borrado. También se trabajó en los objetivos de la memoria.

La lista de tareas está disponible en [Sprint 9¹⁰](https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/10?closed=1). Todas las tareas se completaron a tiempo. El gráfico *burndown* del *sprint* se ve en la figura A.7.

Figura A.7: Burndown del *sprint* 9

¹⁰<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/10?closed=1>

Sprint 10

Este *sprint* se centró en añadir la funcionalidad de minería de datos respecto a creación de clasificadores y varias mejoras visuales de la interfaz.

Este *sprint* duró dos semanas por desarrollarse a la vez que la época de exámenes. Las tareas relacionadas con la mejora de la web no hubo problemas en completarlas, pero las relacionadas a la creación de clasificadores llevaron más tiempo del esperado y no se pudieron completar a tiempo.

La lista de tareas está disponible en [Sprint 10¹¹](#). El gráfico *burndown* del *sprint* se ve en la figura A.8.

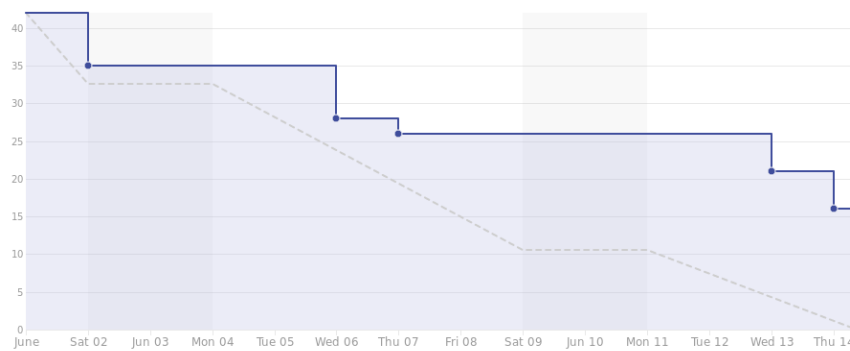


Figura A.8: Burndown del *sprint* 10

Sprint 11

En este *sprint* se completaron las tareas pendientes del *sprint* anterior, además se realizó también la tarea de predecir nuevos espectros. También se avanzó bastante en la memoria, aunque no se llegaron a terminar completamente todas las secciones previstas.

Debido a la cercanía de la entrega este *sprint* tuvo más carga de trabajo que los anteriores. Se puede considerar el último *sprint* de desarrollo porque

¹¹<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/11?closed=1>

se terminaron de implementar los objetivos del proyecto.

La lista de tareas está disponible en [Sprint 11¹²](#). El gráfico *burndown* del *sprint* se ve en la figura A.9. Debido a que gran parte de las tareas eran de documentación, se avanzaba en ellas en paralelo y se cerraron al final del *sprint*, de ahí la forma del gráfico.



Figura A.9: Burndown del *sprint* 11

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

¹²<https://github.com/IvanBeke/TFG-Visor-de-espectros/milestone/12?closed=1>

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apéndice se describen los objetivos generales de la aplicación y se detallan sus requisitos, tanto funcionales como no funcionales.

B.2. Objetivos generales

- Ofrecer control de usuarios.
- Permitir a los usuarios subir *datasets* y espectros.
- Que los *datasets* y espectros se puedan visualizar.
- Que sobre la visualización se pueda aplicar operaciones de procesamiento.
- Poder entrenar modelos de aprendizaje automático con los *datasets* subidos.
- Poder usar los modelos mencionados anteriormente para predecir nuevos ejemplos subidos.
- Que la aplicación final sea útil para la investigadora.

B.3. Catalogo de requisitos

Requisitos funcionales

- **RF-1 Control de usuarios:** la aplicación debe permitir controlar usuarios.
 - **RF-1.1 Integración con Google:** la aplicación debe poder hacer uso de cuentas de Google para el control de usuarios.
 - **RF-1.2 Inicio de sesión:** el usuario debe poder iniciar sesión con una cuenta de Google.
 - **RF-1.3 Cierre de sesión:** el usuario debe poder cerrar sesión cuando haya terminado.
- **RF-2 Datasets:** la aplicación debe poder almacenar y gestionar conjuntos de espectros.
 - **RF-2.1 Subida:** el usuario debe poder subir un *dataset*.
 - **RF-2.2 Eliminado:** el usuario debe poder eliminar un *dataset* almacenado.
 - **RF-2.3 Visualización:** el usuario debe poder visualizar un *dataset* almacenado.
- **RF-3 Espectros:** la aplicación debe poder almacenar y gestionar espectros.
 - **RF-3.1 Subida:** el usuario debe poder subir un espectro.
 - **RF-3.2 Eliminado:** el usuario debe poder eliminar un espectro.
 - **RF-3.3 Visualización:** el usuario debe poder visualizar un espectro.
- **RF-4 Procesamiento:** el usuario debe poder aplicar operaciones de preprocesamiento.
 - **RF-4.1: Procesamiento de *dataset*:** el usuario debe poder aplicar operaciones de preprocesamiento sobre un *dataset* visualizado.
 - **RF-4.2: Procesamiento de espectro:** el usuario debe poder aplicar operaciones de preprocesamiento sobre un espectro visualizado.

- **RF-5 Minería de datos:** el usuario debe poder usar técnicas de minería de datos.
 - **RF-5.1 Creación:** el usuario debe poder crear modelos personalizados.
 - **RF-5.2 Evaluación:** la aplicación debe ofrecer métricas del modelo entrenado.
 - **RF-5.3 Predicción:** el usuario debe poder usar los modelos que ha creado para predecir nuevos espectros.

Requisitos no funcionales

- **RNF-1 Usabilidad:** la aplicación debe ser intuitiva y fácil de usar.
- **RNF-2 Escalabilidad:** el rendimiento debe poder aumentar al aumentar los recursos.
- **RNF-3 Mantenibilidad:** debe ser sencillo añadir funcionalidad nueva a la aplicación.
- **RNF-4 Compatibilidad:** la aplicación debe poder funcionar en los principales navegadores.
- **RNF-5 Responsividad:** la aplicación debe adaptarse al tamaño de la pantalla.
- **RNF-6 Facilidad de despliegue:** la aplicación debe poder desplegarse en un servidor de forma sencilla.

B.4. Especificación de requisitos

En esta sección se presentan el diagrama de casos de uso. A continuación, se desarrollan los casos de uso mostrados en el diagrama.



CU-1	Iniciar sesión
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-1, RF-1.1, RF-1.2
Descripción	El usuario inicia sesión en la aplicación
Precondición	Navegador web abierto y página de la aplicación cargada
Acciones	<ol style="list-style-type: none"> 1. Pulsar botón “Iniciar sesión con Google”. 2. Introducir o seleccionar cuenta de Google.
Postcondición	Redirección a la aplicación con sesión iniciada
Excepciones	<ul style="list-style-type: none"> ■ Cuenta no existente. ■ Combinación de nombre y contraseña incorrecta.
Importancia	Alta

CU-1	Iniciar sesión
-------------	-----------------------

Tabla B.1: Iniciar sesión.

CU-2	Cerrar sesión
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-1, RF-1.3
Descripción	El usuario cierra sesión en la aplicación
Precondición	Sesión iniciada en la aplicación
Acciones	<ol style="list-style-type: none">1. Pulsar en el botón cuyo texto es el correo.2. Pulsar “Cerrar sesión”.
Postcondición	Redirección a la aplicación con sesión cerrada
Excepciones	<ul style="list-style-type: none">■ La sesión no estaba iniciada.
Importancia	Alta

Tabla B.2: Cerrar sesión.

CU-3	Subir dataset
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-2, RF-2.1
Descripción	El usuario sube un dataset a la aplicación para su guardado
Precondición	Sesión iniciada en la aplicación
Acciones	<ol style="list-style-type: none"> 1. Pulsar en el botón “Subir dataset”. 2. Descargar y rellenar la plantilla. 3. Crear un fichero <code>.zip</code> con los datos y la plantilla. 4. Rellenar el formulario de subida. 5. Seleccionar el fichero creado. 6. Presionar el botón “Subir”.
Postcondición	Redirección a la página de los archivos guardados
Excepciones	<ul style="list-style-type: none"> ■ El formato del dataset no es correcto. ■ Existe un dataset con el mismo nombre.
Importancia	Alta

Tabla B.3: Subir dataset.

CU-4	Subir espectro
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-3, RF-3.1
Descripción	El usuario sube un espectro a la aplicación para su guardado
Precondición	Sesión iniciada en la aplicación
Acciones	<ol style="list-style-type: none"> 1. Pulsar en el botón “Subir espectro”. 2. Rellenar el formulario de subida. 3. Seleccionar el espectro. 4. Presionar botón “Subir”.
Postcondición	Redirección a la página de los archivos guardados
Excepciones	<ul style="list-style-type: none"> ■ El formato del espectro no es correcto. ■ Existe un espectro con el mismo nombre.
Importancia	Alta

Tabla B.4: Subir espectro.

CU-5	Visualizar dataset
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-2, RF-2.3, RF-4, RF-4.1
Descripción	El usuario visualiza un espectro en la aplicación
Precondición	Sesión iniciada en la aplicación, dataset guardado en la aplicación y estar en la página de “Mis ficheros”
Acciones	<ol style="list-style-type: none"> 1. Escoger un dataset que visualizar. 2. Pulsar el botón “Visualizar” en el dataset escogido.
Postcondición	Se muestra la página de visualización
Excepciones	<ul style="list-style-type: none"> ■ No se ha podido cargar el dataset.
Importancia	Alta

Tabla B.5: Visualizar dataset.

CU-6	Visualizar espectro
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-3, RF-3.3, RF-4, RF-4.2
Descripción	El usuario visualiza un espectro en la aplicación
Precondición	Sesión iniciada en la aplicación, espectro guardado en la aplicación y estar en la página de “Mis ficheros”
Acciones	<ol style="list-style-type: none"> 1. Escoger un espectro que visualizar. 2. Pulsar el botón “Visualizar” en el espectro escogido.
Postcondición	Se muestra la página de visualización
Excepciones	<ul style="list-style-type: none"> ■ No se ha podido cargar el espectro.
Importancia	Alta

Tabla B.6: Visualizar espectro.

CU-7	Guardar clasificador
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-5, RF-5.1, RF-5.2
Descripción	El usuario crear y entrena un modelo usando como datos un dataset guardado
Precondición	Sesión iniciada en la aplicación, dataset guardado en la aplicación y estar en la página de “Mis ficheros”
Acciones	<ol style="list-style-type: none"> 1. Escoger el dataset que se quiera usar como base. 2. Pulsar el botón “Crear modelo” en el dataset escogido. 3. Seleccionar en el desplegable el modelo a usar. 4. (Opcional) Rellenar el formulario con los parámetros del modelo. 5. Pulsar el botón “Crear y evaluar modelo”. 6. Para guardar el clasificador: <ol style="list-style-type: none"> a) Completar el formulario. b) Pulsar el botón “Guardar”. 7. Para no guardar el clasificador: <ol style="list-style-type: none"> a) Pulsar el botón “Descartar”.
Postcondición	Se redirige a los archivos guardados
Excepciones	<ul style="list-style-type: none"> ■ Los parámetros del modelo introducidos son erróneos. ■ Ya existe un clasificador con el nombre introducido.
Importancia	Media

Tabla B.7: Guardar clasificador.

CU-8	Predecir espectro
Versión	1.0
Autor	Iván Iglesias Cuesta
Requisitos asociados	RF-5, RF-5.3
Descripción	El usuario usa un clasificador creado para predecir un espectro.
Precondición	Sesión iniciada en la aplicación, dataset guardado en la aplicación, algún clasificador creado y estar en la página de “Mis ficheros”
Acciones	<ol style="list-style-type: none"> 1. Escoger el espectro que se quiera predecir. 2. Pulsar el botón “Predecir” en el espectro escogido. 3. Seleccionar en el desplegable el clasificador a usar. 4. Pulsar el botón “Predecir espectro”. 5. Para guardar el clasificador:
Postcondición	Se muestra la predicción
Excepciones	<ul style="list-style-type: none"> ■ No se puede cargar el clasificador.
Importancia	Media

Tabla B.8: Predecir espectro.

Apéndice C

Especificación de diseño

- C.1. Introducción
- C.2. Diseño de datos
- C.3. Diseño procedimental
- C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

D.1. Introducción

D.2. Estructura de directorios

D.3. Manual del programador

Manual de despliegue

El despliegue de la aplicación se hace mediante la herramienta Nanobox, cuyo propósito principal es el de facilitar la tarea de despliegue. Para ello combina servicios de virtualización con servidores virtuales privados o VPS, ocupándose de la configuración de la máquina virtual que el proveedor de VPS nos proporcione.

El primer paso es registrarse en alguno de los proveedores disponibles¹. Para este proyecto se ha elegido Digital Ocean debido a que durante el proyecto se contaba con el “Student Developer Pack” de GitHub, el cual contiene un crédito gratuito de 50\$ para esa plataforma.

El siguiente paso es crear una cuenta en Nanobox². Una vez hecho, hay

¹<https://docs.nanobox.io/providers/hosting-accounts/>

²<https://nanobox.io/>

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

que enlazar esta cuenta con la creada anteriormente, desde las opciones de la cuenta en la pestaña “Hosting Accounts”, una vez ahí seguir las instrucciones que se muestran en la página.

Después de tener las cuentas preparadas hay que descargar la herramienta Nanobox para el sistema adecuado e instalarla, seguir las instrucciones de la documentación oficial³ y las del instalador.

Desde la página principal se pulsa en “Launch New App” para crear la nueva aplicación, seguir las instrucciones en la página. Al terminar se muestran instrucciones para desplegar la aplicación.

Como se puede comprobar, Nanobox cumple el propósito de facilitar el despliegue. Para cualquier otra duda sobre la herramienta toda la documentación oficial está disponible en <https://docs.nanobox.io/>.

D.4. Compilación, instalación y ejecución del proyecto

D.5. Pruebas del sistema

³<https://docs.nanobox.io/install/>

Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuarios
- E.3. Instalación
- E.4. Manual del usuario

Bibliografía
