



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



TFG del Grado en Ingeniería  
Informática

Data warehouse Sistema de  
ayuda para la asignación en  
sigma



Presentado por Álvaro Urdiales Santidrián  
en Universidad de Burgos — 4 de febrero  
de 2020

Tutor: Carlos Pardo Aguilar



---

# Índice general

---

<b>Índice general</b>	<b>I</b>
<b>Índice de figuras</b>	<b>III</b>
<b>Índice de tablas</b>	<b>IV</b>
<b>Apéndice A Plan de Proyecto Software</b>	<b>1</b>
A.1. Introducción . . . . .	1
A.2. Planificación temporal . . . . .	2
A.3. Estudio de viabilidad . . . . .	6
<b>Apéndice B Especificación de Requisitos</b>	<b>11</b>
B.1. Introducción . . . . .	11
B.2. Objetivos generales . . . . .	12
B.3. Catalogo de requisitos . . . . .	12
B.4. Especificación de requisitos . . . . .	15
<b>Apéndice C Especificación de diseño</b>	<b>21</b>
C.1. Introducción . . . . .	21
C.2. Diseño de datos . . . . .	21
C.3. Diseño procedimental . . . . .	23
C.4. Diseño arquitectónico . . . . .	24
<b>Apéndice D Documentación técnica programador</b>	<b>27</b>
D.1. Introducción . . . . .	27
D.2. Estructura de directorios . . . . .	27
D.3. Manual del programador . . . . .	27

<b>Apéndice E Documentación de usuario</b>	<b>35</b>
E.1. Introducción . . . . .	35
E.2. Requisitos de usuarios . . . . .	35
E.3. Instalación . . . . .	35
E.4. Manual del usuario . . . . .	37
<b>Bibliografía</b>	<b>41</b>

---

# Índice de figuras

---

A.1. Clasificación licencias software.[4]	8
B.1. Diagrama casos de uso 1.	14
B.2. Diagrama casos de uso 2.	14
C.1. Diagrama entidad relación	22
C.2. Diagrama relacional	23
C.3. Modelo vista presentador	25
D.1. Instalación anaconda.	29
D.2. Configuración contraseña MySQL.	30
D.3. Instalación cherrypy.	31
D.4. Instalación SQL alchemy.	31
D.5. Configuración conexión MySQL Workbench.	32
D.6. Importar datos en MySQL.	33
D.7. Archivo a importar en MySQL.	33
E.1. Compilación y ejecución del proyecto.	36
E.2. Vista del proyecto en el navegador.	36
E.3. Introducir datos nuevos en la aplicación.	37
E.4. Selección de archivo para la actualización de datos.	39
E.5. Menú actualización de datos.	39
E.6. Menú visualización de datos.	39
E.7. Menú selección de grado.	40

---

# Índice de tablas

---

A.1. Preguntas planificación proyecto . . . . .	1
A.2. Costes programador . . . . .	6
A.3. Costes hardware . . . . .	6
A.4. Costes software . . . . .	7
A.5. Costes varios . . . . .	7
A.6. Total costes . . . . .	7
A.7. Licencias Software . . . . .	8
B.1. Gestión de base de datos . . . . .	15
B.2. Visualizar datos . . . . .	16
B.3. Introducir datos . . . . .	17
B.4. Actualizar base de datos . . . . .	18
B.5. Borrar base de datos . . . . .	19

## Apéndice A

---

# Plan de Proyecto Software

---

### A.1. Introducción

Antes de comenzar cualquier proyecto, es muy recomendable tomarse un tiempo en planificarlo. Por ello, la planificación es un punto clave en cualquier proyecto, el cual consiste en organizar y racionalizar aquello que queremos hacer alcanzando una serie de determinados objetivos. En esta etapa, se analizarán cada una de las partes en las que se compone el proyecto, permitiendo así conocer los recursos necesarios. Para la planificación de un proyecto, es bueno que nos hagamos una serie de preguntas, cuyas respuestas nos ayudarán con la toma de decisiones a la hora de tomar o descartar propuestas y organizarnos.

	Pregunta	Respuesta
Qué	Quieres hacer	Descripción y finalidad
Por qué	Lo quieres hacer	Fundamentación
Para qué	Se quiere hacer	Objetivos
Cuánto	Quieres conseguir	Metas
Dónde	Se quiere hacer	Localización física y cobertura espacial
Cómo	Se va a hacer	Actividades y tareas Metodología
Cuándo	Se va a hacer	Calendario
A quiénes	Va dirigido	Destinatarios o beneficiarios
Quiénes	Lo van a hacer	Recursos humanos
Con qué	Se va a hacer o se va a costear	Recursos materiales y financieros

Tabla A.1: Preguntas planificación proyecto

La fase de planificación, podemos dividirla en dos partes:

- Planificación temporal: en la que se estimarán los costes en cuanto a tiempo de cada una de las partes del proyecto, estableciendo una fecha de inicio y una fecha estimada de finalización.
- Estudio de viabilidad: centrada en la viabilidad del proyecto, la cual podemos dividirla en dos partes:
  - Viabilidad económica: centrada en costes y beneficios del proyecto.
  - Viabilidad legal: centrada en la parte legislativa del proyecto, como pueden ser las leyes que afectan al software, licencias y ley de protección de datos.

## A.2. Planificación temporal

Para la planificación del proyecto, se intentó seguir una de las metodologías dadas durante la carrera, la cual es utilizada en numerosas empresas hoy en día y que ayuda a reducir la complejidad en el desarrollo, satisfaciendo las necesidades del cliente promoviendo la colaboración.

Es por ello que durante el proyecto se trato de seguir estas tecnologías ágiles, en concreto Scrum, aunque debido a ser un proyecto educativo y al haber unicamente una persona trabajando en el, no se pudo seguir al pie de la letra.

Para ello, se siguieron una serie de pautas basadas en esta metodología:

- Seguimiento del proceso mediante *Sprints*, cuya duración seria de una semana.
- Desarrollo incremental en el que se realizaban *Sprints* y revisiones.
- En las reuniones de finalización e inicio de los *Sprints*, se revisaba el producto entregado y se planificaba el nuevo *Sprint*, otorgando una serie de tareas a realizar.

### Sprint 0

Este *sprint* marco el inicio del proyecto. Anteriormente ya se me había planteado la idea del proyecto aunque fue a rasgos generales, por lo que en



este *sprint*, se hablo en mayor profundidad del proyecto, profundizando en los requisitos y objetivos del mismo.

Por otro lado, se estableció un orden a rasgos generales en las tareas y de posibilidades y herramientas a la hora de realizarlo, descartando algunas ideas y generando tareas de investigación en otras, debido a su posible validez para la realización del mismo.

## Sprint 1

Los objetivos de este *sprint* consistían en decidir las herramientas que se utilizarían en el proyecto una vez investigadas las posibilidades. Fue aquí donde se descartaron las bases de datos no relacionales y se decidió ahorrar costes eligiendo una base de datos gratuita.

Como tareas se propuso el diseño conceptual de la base de datos teniendo en cuenta los datos que íbamos a manejar y que esta debería estar preparada para soportar gran cantidad de datos y la investigación del sistema gestor de la base de datos propuesto para albergarlo.

## Sprint 2

Los objetivos de este *sprint* fueron poner en común el diseño de la base de datos buscándola fallos y posibles correcciones, creando como tarea el paso del diseño conceptual al lógico y posteriormente implantarlo en el sistema gestor de la base de datos, creando asimismo una función que permita la restauración de la base de datos borrando todos los datos y dejando la estructura vacía.

Por otro lado tenía como tarea extra investigar la forma en la que trataría los datos internamente en el proyecto, que estructuras los recogerían para su tratamiento antes de insertarlos en la base de datos.

## Sprint 3

El objetivo de este *sprint* consistió en la depuración de la base de datos creada, comentando la forma en la que se leerían los datos y se insertarían en ella una vez habían sido estudiados.

Como tarea se me dio el archivo a leer para que investigara la forma en la que lo leería y procesaría, para su posterior subida a la base de datos.

## Sprint 4

En el se comentaron problemas en cuanto a la lectura del archivo, debido a que este está corrupto y la forma de la que se tenía pensado leerlo pasaba a ser inválida. También se hablo de la estructura decidida para el tratamiento de datos y la forma de la que se subiría a la base de datos.

Como solución se propuso la creación de un parse que leyera el archivo en modo texto recogiendo los datos relevantes para su posterior almacenamiento y tratamiento.

## Sprint 5

En este *sprint* se comprobó que el parse creado funcionaba correctamente, recogiendo los datos y guardándolos en la estructura que los trataría para su posterior inserción en la base de datos.

Como tarea se propuso la subida de todos estos datos leídos a la base de datos, estudiando el uso de una posible interfaz que hiciera este proceso a ojos del cliente mas sencillo.

## Sprint 6

El objetivo de este *sprint* consistió en comprobar el correcto funcionamiento de la subida de datos a la base de datos, viendo también el prototipo de interfaz que tendría el programa. Una vez comprobado el correcto funcionamiento del mismo, se mandaron las tareas entre las que se encontraban una depuración del mismo, junto con una pequeña etapa de investigación para la actualización de los datos.

## Sprint 7

El objetivo de este *sprint* consistió en una lluvia de ideas una vez estudiado el problema de la actualización sobre como comprobar que datos están en la base de datos y cuales hay que sustituir, así como comprobar que la lectura y subida de datos funcionaba correctamente.

Como tarea, se propuso la implementación de la actualización de datos en el proyecto.

## Sprint 8

En este *sprint* se comprobó el correcto funcionamiento de la actualización de datos, y se propuso como tarea mostrar en el proyecto que datos se van a actualizar, así como la implementación de la interfaz gráfica de esta parte y una pequeña depuración en la actualización, puesto que en algunos casos daba errores.

## Sprint 9

En este *sprint* se comprobó el correcto funcionamiento de la interfaz, por otro lado el tutor me sugirió mostrar en lugar de una tabla donde se mostraran los datos a actualizar, dos tablas; una con los datos obsoletos de la base de datos, osea aquellos que se van a borrar y otra con los datos nuevos del excel, osea aquellos que corresponderían con los datos nuevos a introducir.

## Sprint 10

El objetivo de este *sprint*, consistía principalmente en comprobar el correcto funcionamiento de la actualización de los datos.

Debido a que únicamente quedaba la parte de visualizar los datos, el tutor me propuso que generara una serie de filtros genéricos, los cuales iríamos adaptando según las necesidades que se tendrían sobre los datos de la base de datos.

## Sprint 11

En este *sprint* se puso en común la idea del prototipo de interfaz para la visualización de datos, así como los filtros genéricos, a los cuales el tutor propuso una serie de arreglos para adaptarles mejor a las necesidades que tenía, así como arreglos menores para una mejora en la visualización de los mismos. Por otro lado, se me pidió que investigara en la forma en la que entregaría el proyecto, enviándole a el un prototipo para probarlo.

## Sprint 12

En este *sprint* se realizaron todas las tareas anteriores, dando por en un principio terminada la parte de programación. Se propuso como tarea el inicio de la documentación del proyecto.

### A.3. Estudio de viabilidad

#### Viabilidad económica

En esta sección se calcularán los costes del proyecto si este hubiera sido realizado en un entorno empresarial en el que se hubiera tenido que pagar por los recursos utilizados.

#### Costes programador

Actualmente un contrato de trabajo hacia un ingeniero informático ronda los 20.000€ anuales brutos, lo que se queda en unos 1366€ netos al mes.

Por lo tanto:

Concepto	Coste
Sueldo Anual	20.000€
Retenciones IRPF	2.338€
Cuotas seguridad social	1.270€
Sueldo neto anual	16.392€
Sueldo mensual	1.666,67€
Sueldo neto mensual	1.366€
<b>Total 6 meses</b>	<b>10.000€</b>

Tabla A.2: Costes programador

#### Costes Hardware

Actualmente, según la agencia tributaria, un equipo para procesos de información se amortiza en un máximo de 8 años y este ha sido utilizado durante 6 meses únicamente para el proyecto.

Por lo tanto:

Concepto	Coste	Coste amortizado
Ordenador portátil	1.069€	66,81€

Tabla A.3: Costes hardware

### Costes software

En esta sección se recogerán todos los posibles costes en cuanto a software se refiere <sup>1</sup>:

Concepto	Coste
Licencia Windows	145€
MySQL	0€
Entorno desarrollo	0€
<b>Total</b>	145€

Tabla A.4: Costes software

### Costes varios

Costes menores del proyecto:

Concepto	Coste	Coste 6 meses
Lugar trabajo	125€	750€
Internet	38€	228€
Impresión trabajo		50€
<b>Total</b>		1.028€

Tabla A.5: Costes varios

### Total costes

La suma de todos los costes anteriores se quedaría en:

Concepto	Coste
Programador	10.000€
Hardware	66,81€
Software	145€
Varios	1.028€
<b>Total</b>	11.239,81€

Tabla A.6: Total costes

---

<sup>1</sup>Se intentó buscar entornos de desarrollo gratuitos ofreciendo así la posibilidad de explotación gratuita del proyecto.

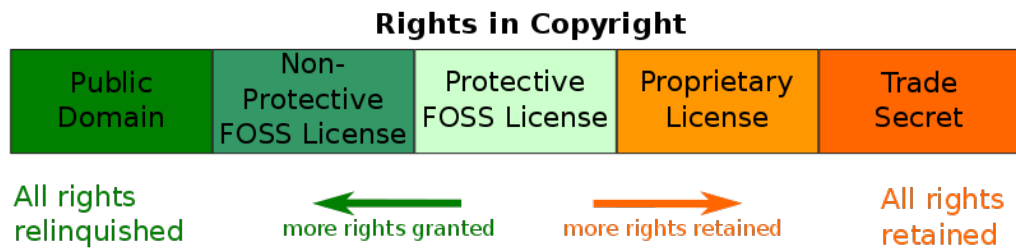


Figura A.1: Clasificación licencias software.[4]

## Viabilidad legal

En esta sección se verán las licencias que poseen todos los elementos utilizados en el desarrollo del proyecto. Incluyendo tanto el propio proyecto como la documentación del mismo. Esto es debido al gran número de licencias que hay y a las restricciones que posee cada una, debiendo elegir una compatible con todas ellas para el proyecto [A.1](#).

Podemos definir una licencia software como el contrato entre el autor o titular de los derechos de explotación y el usuario que deberá cumplir una serie de reglas o normas para la utilización del producto.[6]

## Viabilidad software

Para el desarrollo del proyecto software se utilizaron los siguientes elementos:

Dependencia	Descripción	Licencia
Anaconda/Spyder	Entorno de desarrollo del proyecto	MIT
CherryPy	Framework orientado a objetos web para Python	BSD
Pandas	Biblioteca Python para la manipulación y análisis de datos	BSD
NumPy	Biblioteca Python que agrega mayor soporte para vectores y matrices	BSD
SQLAlchemy	<i>Toolkit</i> SQL para Python	MIT
MySQL	Sistema gestor de base de datos	GNU

Tabla A.7: Licencias Software

Las licencias BSD y MIT, son las más permisivas, mientras que la GNU

es más restrictiva obligando que todas las modificaciones sobre el software original deben realizarse bajo la misma licencia.

Por otro lado debido a la incompatibilidad de las licencias BSD con GNU, el proyecto poseerá dos tipos de licencias. Por un lado la parte de la base de datos se encontrará bajo una licencia GNU, mientras que el resto del proyecto se encontrará bajo una licencia BSD. [1]





## *Apéndice B*

---

# Especificación de Requisitos

---

## B.1. Introducción

La especificación de requisitos de software (ERS) consiste en una descripción del comportamiento de la herramienta que se va a desarrollar. Esto contiene el conjunto de casos de uso (o requisitos funcionales) que describirán todas las posibles acciones o actividades que los usuarios podrán realizar sobre el proyecto.

Todo ello está dirigido tanto al cliente como al equipo de desarrollo y debe ser fácilmente comprensible para todas las partes involucradas en el desarrollo.<sup>[5]</sup>

Según el estándar IEEE 830-1998, una ERS debe cumplir las siguientes características:

- **Completa:** Todos los requerimientos deben estar reflejados en ella y todas las referencias deben estar definidas.
- **Consistente:** Debe ser coherente con los propios requerimientos y también con otros documentos de especificación.
- **Inequívoca:** La redacción debe ser clara de modo que no se pueda mal interpretar.
- **Correcta:** El software debe cumplir con los requisitos de la especificación.

- Trazable: Se refiere a la posibilidad de verificar la historia, ubicación o aplicación de un ítem a través de su identificación almacenada y documentada.
- Priorizable: Los requisitos deben poder organizarse jerárquicamente según su relevancia para el negocio y clasificándolos en esenciales, condicionales y opcionales.
- Modificable: Aunque todo requerimiento es modificable, se refiere a que debe ser fácilmente modificable.
- Verificable: Debe existir un método finito sin costo para poder probarlo.

## B.2. Objetivos generales

El proyecto ha intentado cumplir los siguientes objetivos generales:

- Dadas una serie de hojas de calculo (.xlsx), deberá permitir importarlas analizando y asignando los datos al departamento, y titulación a la que pertenecen los diferentes datos.
- Si se da el caso de que la información importada es una actualización de la información existente, deberá comparar ambas informaciones, mostrando las diferencias y dando la opción de sobrescribir los datos antiguos por los nuevos o desechar lo nuevo.
- Una vez importados los datos, deberá permitir filtrar toda la información que posea para su posterior búsqueda.
- Tras esta búsqueda de información, deberá mostrar la información buscada permitiendo una mejor comprensión de la misma.

## B.3. Catalogo de requisitos

A partir de los objetivos generales, se han desarrollado una serie de requisitos mas específicos:

### Requisitos funcionales

- **RF-1 Gestionar Base de datos:** el proyecto tiene que ser capaz de gestionar una gran base de datos donde se encuentra toda la información.

- **RF-1.1 Añadir nuevos datos:** el usuario debe poder añadir nuevos datos desde un excel a la base de datos
  - **RF-1.1.1 Lectura de datos:** los datos introducidos pueden ser corruptos pero a la hora de la lectura debe ser irrelevante para el usuario.
- **RF-1.2 Actualizar datos:** el usuario debe poder actualizar datos ya existentes dentro de la base de datos.
  - **RF-1.2.1 Datos ya actualizados:** en caso de que los datos a actualizar ya estén en la base de datos, el programa lanzará un aviso cancelando así la actualización de los mismos.
  - **RF-1.2.2 Datos nuevos:** en caso de que los datos no estén en la base de datos, el programa lanzara un aviso para que se introduzcan, ya que actualizar no seria la opción correcta.
  - **RF-1.2.3 Datos actualizables:** en caso de que los datos sean actualizables, el programa ofrecerá una opción para poder ver los datos que se van a actualizar y los datos obsoletos.
- **RF-1.3 Borrar base de datos:** el usuario podrá reseatear la base de datos, borrando todos los datos de ella pero dejando la estructura de la misma.
- **RF-2 Visualización de datos:** el usuario podra visualizar los datos de la base de datos filtrados para una mejor comprensión.
  - **RF-2.1 Filtro por carreras:** debido a que la base de datos alberga datos de numerosas carreras, a la hora de visualizar los datos el programa deberá mostrar los diferentes grados de los que contiene información permitiendo al usuario seleccionar de que grado es del que se desea mostrar la información.

### Requisitos no funcionales

- **RNF-1 Usabilidad:** el programa deberá poseer una interfaz sencilla e intuitiva la cual no necesite de un manual para su utilización.
- **RNF-2 Capacidad y Escalabilidad:** la base de datos debe estar preparada para acoger gran cantidad de datos, permitiendo añadir nuevas funcionalidades fácilmente.
- **RNF-3 Rendimiento:** se debe buscar los menores tiempos de carga al conectar con la base de datos, haciendo uso de las posibilidades que esto ofrece (vistas, funciones, etc.)

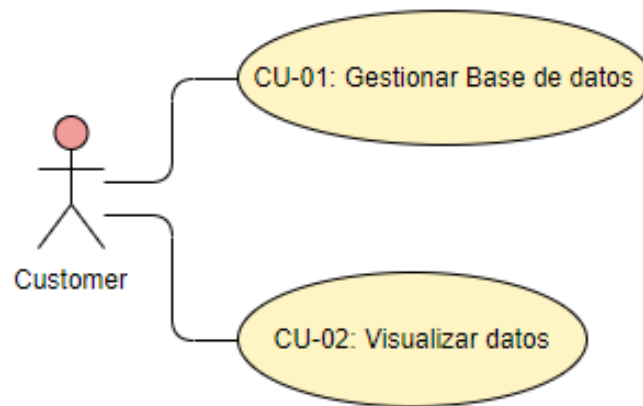


Figura B.1: Diagrama casos de uso 1.

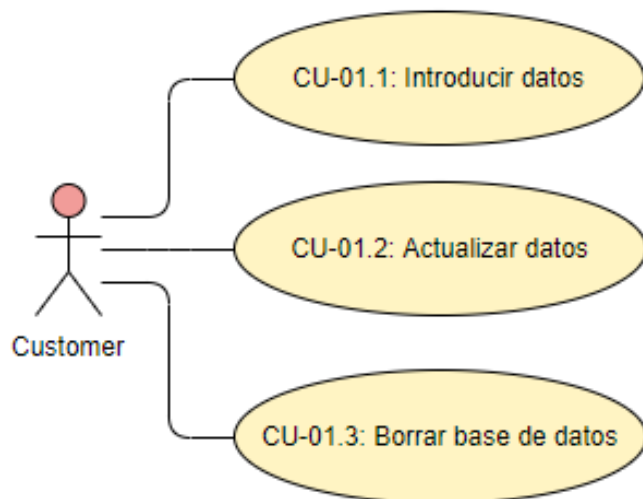


Figura B.2: Diagrama casos de uso 2.

- **RNF-4 Modularidad:** se ha intentado realizar el proyecto de la forma más independiente posible, permitiendo así mayor facilidad a la hora de mantener, modificar, escribir y depurar.

## B.4. Especificación de requisitos

En esta sección es donde se desarrollará el diagrama de casos de uso de las figuras B.1 y B.2, desarrollando cada uno de ellos.

CU-01	Gestión de base de datos
Descripción	Permite gestionar la base de datos de forma sencilla y visual para el usuario.
Precondición	<ul style="list-style-type: none"> <li>■ La base de datos debe estar disponible y accesible.</li> <li>■ Para la modificación o introducción de datos, el archivo debe encontrarse en la misma ruta que el proyecto.</li> </ul>
Acciones	<p>Se mostrará un menu de inicio donde el usuario podrá seleccionar si desea:</p> <ol style="list-style-type: none"> <li>1. Introducir nuevos datos.</li> <li>2. actualizar datos.</li> <li>3. borrar base de datos.</li> </ol>
Postcondición	El programa mostrará un mensaje para hacer ver al usuario que se ha realizado correctamente la acción seleccionada.
Excepciones	-
Requisitos asociados	RF-1, RF-1.1, RF-1.1.1, RF-1.2, RF-1.2.1, RF-1.2.2, RF-1.2.3, RF-1.3

Tabla B.1: Gestión de base de datos

<b>CU-02</b>	<b>Visualizar datos</b>
Descripción	Permitirá la visualización de los datos de la base de datos de forma sencilla y ordenada.
Precondición	La base de datos debe estar disponible y accesible.
Acciones	<ol style="list-style-type: none"> <li>1. El usuario seleccionará esta opción en el menú de inicio.</li> <li>2. Se dará a elegir entre una serie de filtros tales como: asignaturas de un grado, profesores por asignatura, profesores por grupo y horas que tiene cada profesor.</li> <li>3. Una vez seleccionado el filtro, el programa listará los diferentes grados existentes en la base de datos con su respectivo código de grado y deberemos seleccionar uno para que la información corresponda con el grado deseado.</li> </ol>
Postcondición	-
Excepciones	En el caso de que se introduzca un código no existente, se mostrará la tabla vacía.
Requisitos asociados	RF-2, RF-2.1

Tabla B.2: Visualizar datos

CU-01.1	Introducir datos
Descripción	Permite introducir nuevos datos en la base de datos desde un excel.
Precondición	<ul style="list-style-type: none"><li>■ La base de datos debe estar disponible y accesible.</li><li>■ Para la correcta lectura de los datos, el archivo debe encontrarse en la misma ruta que el proyecto.</li></ul>
Acciones	<ol style="list-style-type: none"><li>1. El usuario seleccionará la opción de introducción de datos.</li><li>2. En la nueva ventana seleccionará el archivo a leer.</li></ol>
Postcondición	El programa mostrará un mensaje para hacer ver al usuario que se han introducido los datos correctamente en la base de datos.
Excepciones	<ul style="list-style-type: none"><li>■ En caso de no encontrarse los datos en la misma ruta que el programa lanzará un mensaje de error.</li><li>■ En caso de existir algún error en la inserción de datos, mostrará por pantalla la tabla en la que no han sido introducidos los datos.</li></ul>
Requisitos asociados	RF-1.1, RF-1.1.1

Tabla B.3: Introducir datos

CU-01.2	Actualizar base de datos
Descripción	Permite modificar o actualizar los datos de la base de datos.
Precondición	<ul style="list-style-type: none"> <li>■ La base de datos debe estar disponible y accesible.</li> <li>■ Para la correcta lectura de los datos, el archivo debe encontrarse en la misma ruta que el proyecto.</li> </ul>
Acciones	<ol style="list-style-type: none"> <li>1. El usuario seleccionará la opción de actualizar datos.</li> <li>2. En la nueva ventana seleccionará el archivo a leer.</li> <li>3. Una vez leído el archivo el usuario podrá actualizarlo directamente o visualizar los cambios encontrados con respecto al original para asegurarse del cambio.</li> </ol>
Postcondición	El programa mostrará un mensaje para hacer ver al usuario que se han actualizado los datos correctamente.
Excepciones	<ul style="list-style-type: none"> <li>■ En caso de no encontrarse los datos en la misma ruta que el programa lanzará un mensaje de error.</li> <li>■ En caso de existir algún error en la actualización de datos, mostrará por pantalla la tabla en la que no han sido actualizados los datos.</li> <li>■ En caso de no haber datos a actualizar puesto que corresponden con los ya existentes, el programa lo avisará.</li> <li>■ En caso de no ser necesaria una actualización, sino una inserción de datos, el programa lo indicará.</li> </ul>
Requisitos asociados	RF-1.2, RF-1.2.1, RF-1.2.2, RF-1.2.3

Tabla B.4: Actualizar base de datos



<b>CU-01.3</b>	<b>Borrar base de datos</b>
Descripción	Permite borrar todos los datos de la base de datos dejando únicamente la estructura de la misma.
Precondición	La base de datos debe estar disponible y accesible.
Acciones	Se seleccionará la opción de borrado de base de datos en el menú de inicio.
Postcondición	El programa mostrará un mensaje cuando la base de datos haya sido borrada.
Excepciones	-
Requisitos asociados	RF-1.3

Tabla B.5: Borrar base de datos



## *Apéndice C*

---

# Especificación de diseño

---

### C.1. Introducción

En este apartado se describirá la forma de resolución de los objetivos y especificaciones que se propusieron con anterioridad. Para ello definiremos tanto los datos que se van a manejar como la arquitectura de los mismos, la interfaz, etc.

### C.2. Diseño de datos

La base de datos cuenta con las siguientes entidades:

- **Grados:** posee tanto el código del grado como el nombre de todos los grados de los que tengamos información.
- **Asignaturas:** posee todas las asignaturas de todos los grados con la información genérica de cada una de ellas. Cada asignatura estará ligada a un grado.
- **Grupos:** posee la información más específica de cada uno de los grupos que hay en cada asignatura. Cada grupo estará ligado a una asignatura.
- **Listados:** posee información sobre los datos de la base de datos como el curso académico y la fecha en la que se obtienen los datos. Esta información estará ligada a un grado, ya que la información se va obteniendo grado a grado.

- **Profesores:** contiene la información de cada profesor, no está ligada a ningún otro atributo, ya que es una entidad independiente.
- **Horas asignadas:** contiene la información sobre las horas que tiene cada profesor en cada grado, grupo y curso escolar, por lo que esta relacionada con todas las tablas.

## Diagrama E/R

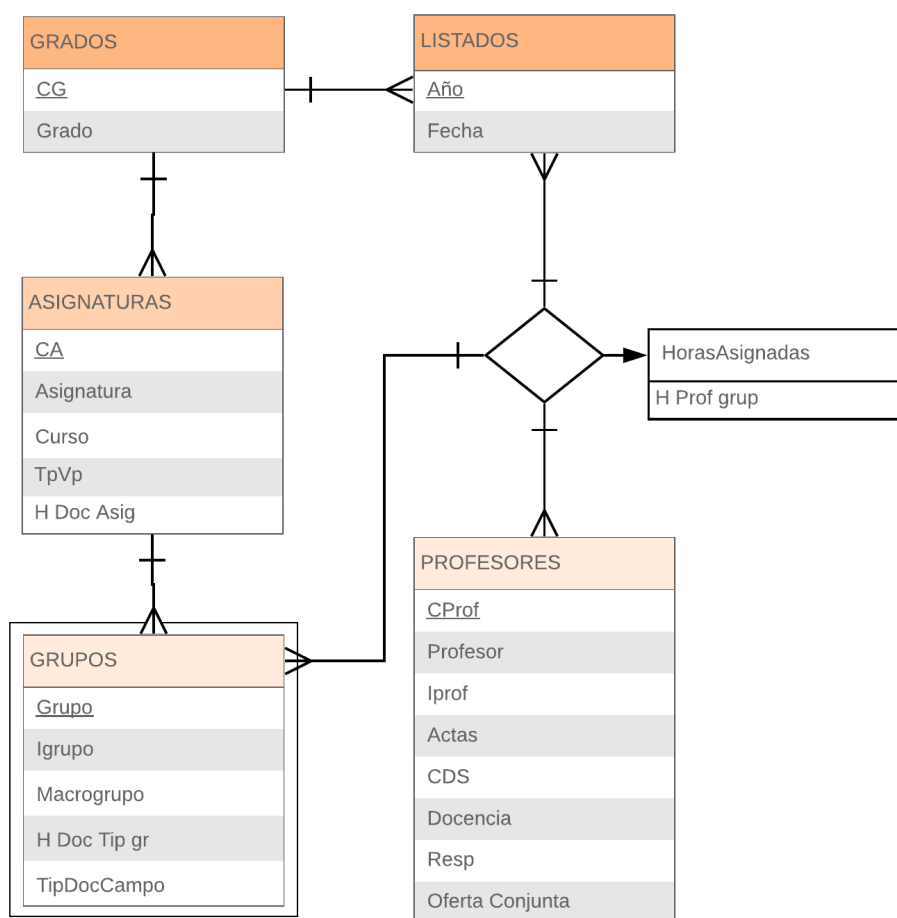


Figura C.1: Diagrama entidad relación

## Diagrama Relacional

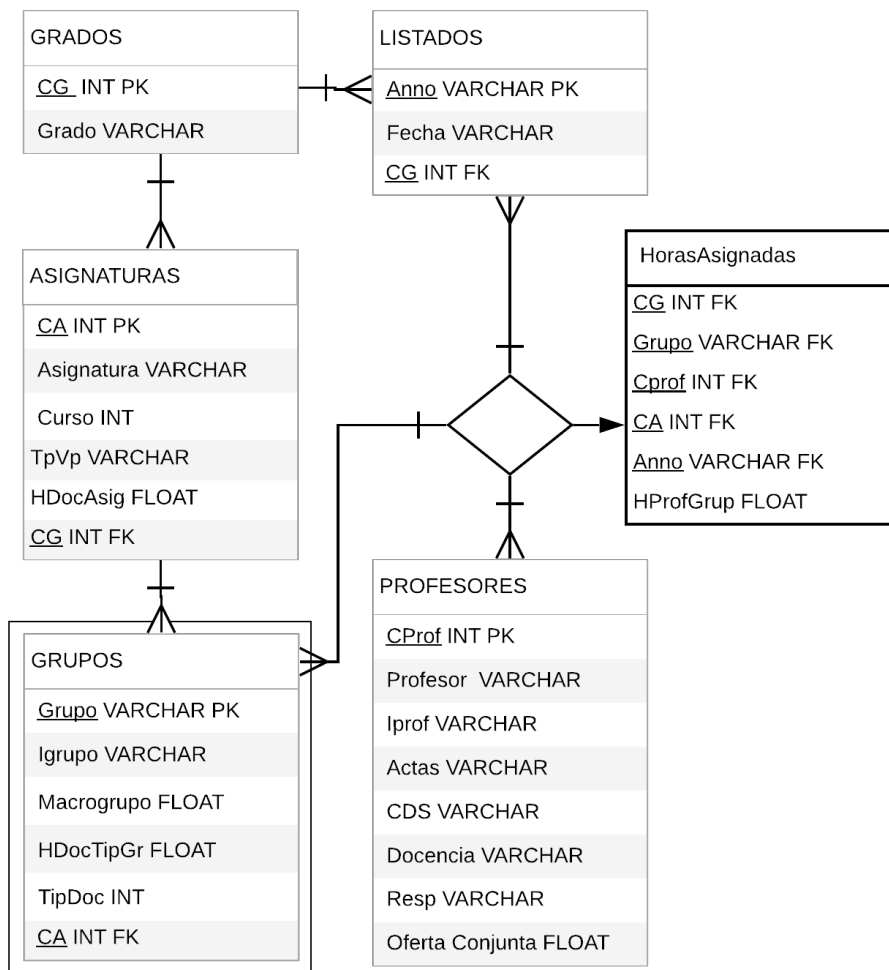


Figura C.2: Diagrama relacional

## C.3. Diseño procedimental

En este apartado se explicarán las funciones principales del proyecto.

### `leerExcel(self, name)`

En esta función, se pasa por parámetro el nombre del excel a leer.

Debido a que los excels se encuentran corruptos, esta función es un parse, el cual irá leyendo el excel en modo texto recogiendo todos los datos relevantes generando un csv en el fichero temporal para su posterior lectura y procesamiento.

**formateoDatos(self)**

Esta función será la encargada de dar un formato adecuado a los diferentes datos del csv, guardándolos en un dataframe el cual nos será devuelto para procesarlo y prepararlo para la base de datos.

**comprobacion(self,dfx,dfy)**

Esta función comprobará los datos de dos dataframes(dfx y dfy), devolviéndonos dos dataframes con los datos distintos que se encuentran en cada uno.

Por un lado nos dará el dataframe borrar, con los datos de dfx que no se encuentran en dfy, y por otro lado nos dará df con los datos de dfy que no están en dfx.

**conexionDB(self)**

Esta función es la encargada de crear una conexión con la base de datos.

## C.4. Diseño arquitectónico

dividir el código en archivos Debido a que el proyecto ha sido realizado en un framework para poder acoplar la interfaz al código, ha condicionado la forma en la que se ha realizado este diseño. A pesar de ello se ha intentado realizar de la manera mas modular posible, permitiendo así mayor facilidad a la hora de mantener, modificar, escribir y depurar.

### Modelo-vista-presentador MVP no

Patrón de diseño derivado del modelo vista controlador utilizado principalmente para la construcción de interfaces de usuario. Es similar al modelo-vista-controlador, con la diferencia de que el controlador desaparece dando paso al presentador, el cual es la capa intermedia la cual posee la lógica de presentación.

Este patrón facilita las pruebas de unidad automatizada y separa los conceptos, permitiendo una simplificación en el desarrollo y mantenimiento de los programas.

Posee 3 componentes distintos, los cuales interactúan entre ellos:[\[7\]](#).

- Modelo: encargado de almacenar y tratar los datos.

- Vista: encargada de la visualización del modelo y órdenes del usuario para actuar sobre los datos.
- Presentador: intermediario entre la vista y el modelo que actúa sobre ellos. Recupera los datos del modelo, tratándolos para mostrarlos en la vista.

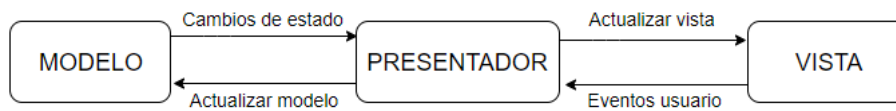


Figura C.3: Modelo vista presentador

Para el uso de este patrón arquitectónico en python, se ha utilizado un framework llamado cherryPy, el cual permitía la integración de una interfaz web para el proyecto.





## *Apéndice D*

---

# Documentación técnica programador

---

### D.1. Introducción

En este apartado se describirá la documentación técnica del proyecto, así como la estructura del proyecto, como su compilación, y la configuración de los diferentes elementos utilizados para su desarrollo.

### D.2. Estructura de directorios

El proyecto está distribuido de la siguiente manera:

- `/`: Contiene el archivo principal del proyecto, el archivo README y la configuración del servidor local del proyecto para su ejecución.
- `/func/`: Contiene el archivo `functions.py`, donde se encuentran las funciones del proyecto y el archivo `presentation.py`, donde encontraremos funciones HTML de la capa presentación del proyecto.
- `/tmp/`: En esta carpeta se almacenarán los archivos temporales que generará la aplicación.

### D.3. Manual del programador

Este apartado será el encargado de ayudar a los futuros programadores con la preparación del entorno de desarrollo, explicando cómo montarlo,

obtener el código fuente del mismo, compilarlo y ejecutarlo.

## Descarga del entorno de desarrollo

El entorno de desarrollo podemos dividirlo en varias partes:

- Programa principal: es el encargado del manejo de la base de datos, preparación de datos y mostrarlos al usuario.
- Base de datos: la encargada de almacenar de forma ordenada todos los datos con los que contará la aplicación.
- Git: encargado del control de versiones.

A continuación se explicarán los pasos a seguir para la instalación de cada uno de los programas.

### Spyder

Spyder será el entorno elegido para el desarrollo del proyecto.

Se podría considerar como un poderoso entorno científico escrito en Python, para Python, y diseñado por y para científicos, ingenieros y analistas de datos. Ofrece una combinación única de la funcionalidad avanzada de edición, análisis, depuración y creación de perfiles de una herramienta de desarrollo integral con la exploración de datos, ejecución interactiva, inspección profunda y hermosas capacidades de visualización de un paquete científico.[3]

Para su instalación bastará con dirigirnos a su web [spyder-ide.org](https://spyder-ide.org) y dirigirnos al apartado downloads, donde nos redirigirá a la web de anaconda, desde la cual seleccionaremos nuestro sistema operativo, la versión de python (En nuestro caso la 3.7) y la version a descargar (64 bit o 32 bit) [D.1](#).

Una vez descargado el archivo bastará con seguir los pasos para su instalación.

### MySQL

MySQL Workbench es el entorno elegido para el alojamiento ordenado de todos los datos.

Se podría considerar como una herramienta visual unificada para arquitectos de bases de datos, desarrolladores y DBA. MySQL Workbench

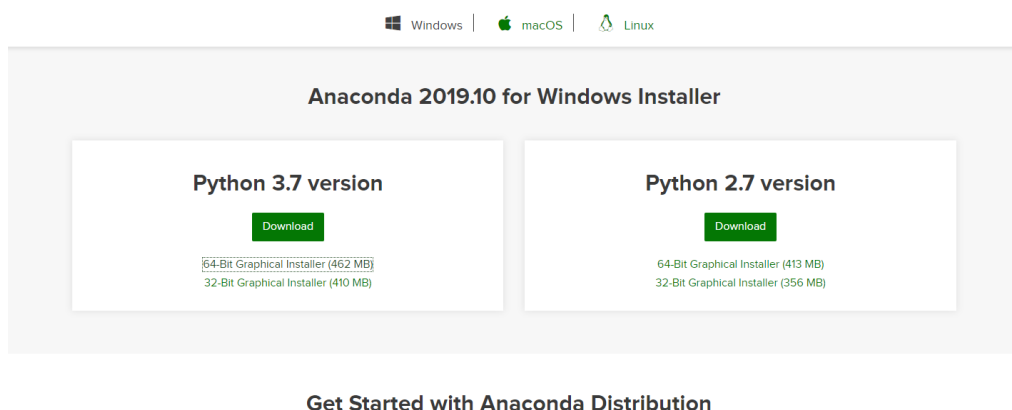


Figura D.1: Instalación anaconda.

proporciona modelado de datos, desarrollo de SQL y herramientas de administración integrales para la configuración del servidor, la administración de usuarios, la copia de seguridad, ...[2]

Para su instalación deberemos dirigirnos a la web e instalar **MySQL**., donde tras dar a descargar deberemos seguir los pasos del instalador. Durante el proceso de instalación elegiremos la instalacion *custom*, y comprobaremos los requisitos para la instalación (sobre todo que se cumplan para MySQL Workbench). Una vez realizado, daremos a next y dejaremos que se vaya instalando todo.

Una vez instalado, nos pedirá una serie de datos para su configuración. A la hora de seleccionar contraseña **D.2** pondremos 1234, ya que el proyecto está configurado para acceder con esa contraseña.

## GitHub Desktop

Para el control de versiones de esta herramienta, he utilizado su versión de escritorio, permitiéndonos así ir subiendo los cambios realizados en el proyecto según se iban realizando.

Para su descarga bastará con acceder a us web **GitHub Desktop**, y descargar el ejecutable, siguiendo los pasos de instalación.

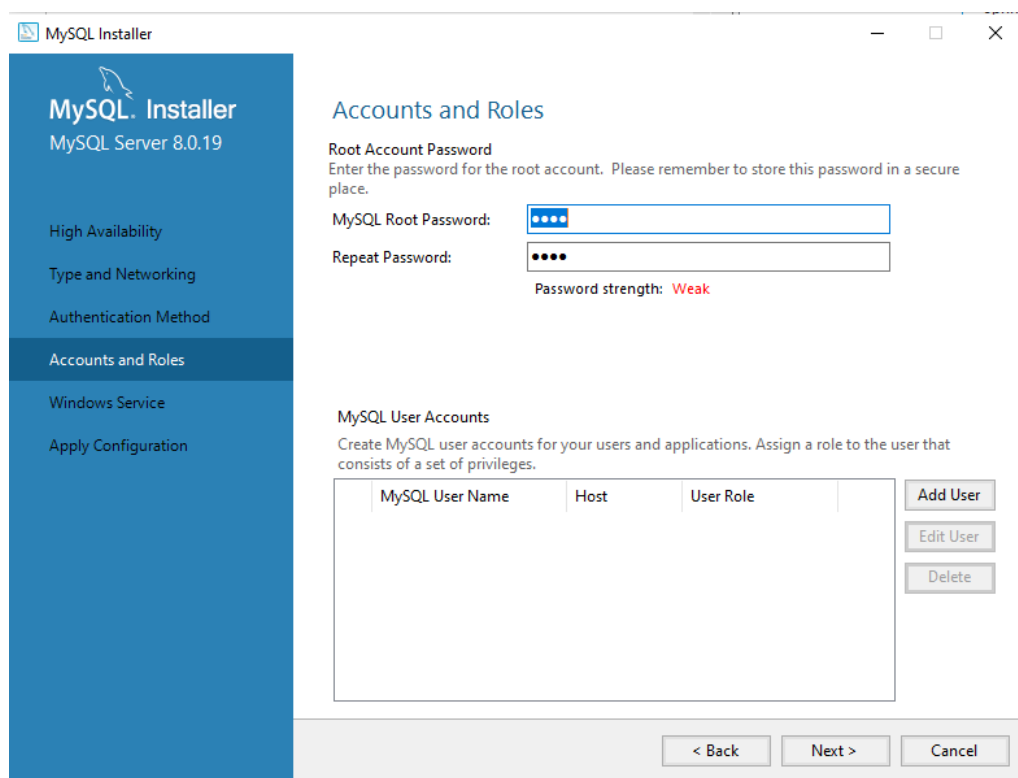


Figura D.2: Configuración contraseña MySQL.

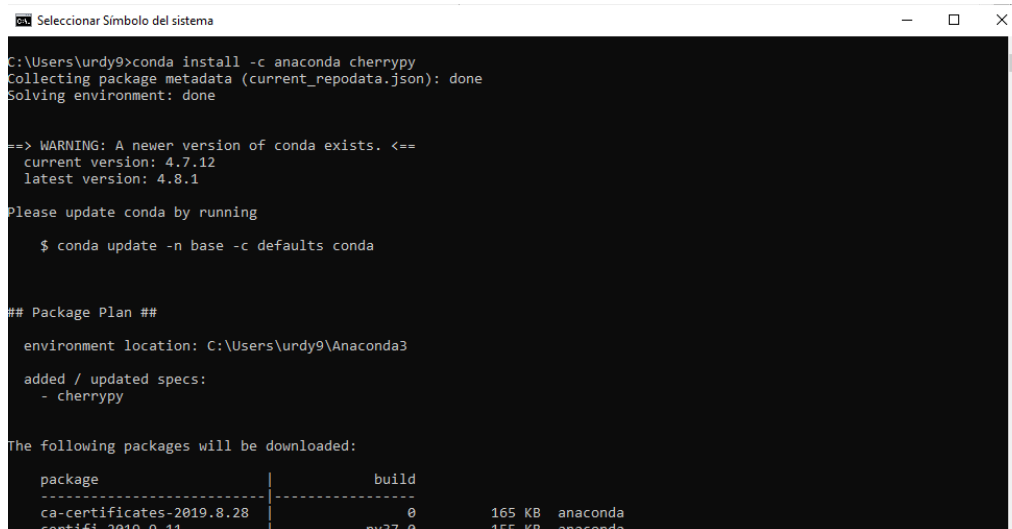
## Configuración del entorno de desarrollo

### Spyder

Una vez instalado el entorno de desarrollo, deberemos instalar el framework para poder compilar el programa. Para poder instalar ese paquete deberemos introducir en la consola de comandos: `conda install -c anaconda cherryypy` [D.3](#).

También deberemos instalar el módulo `sql alchemy` para anaconda, mediante el comando: `conda install -c anaconda sqlalchemy` [D.4](#)

Después de instalarlo, cogeremos los archivos del programa y tras ponerlos en la ruta deseada, estarán listos para ser utilizados.



```
Selecconar Símbolo del sistema
C:\Users\urdy9>conda install -c anaconda cherry
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
current version: 4.7.12
latest version: 4.8.1

Please update conda by running

$ conda update -n base -c defaults conda

## Package Plan ##

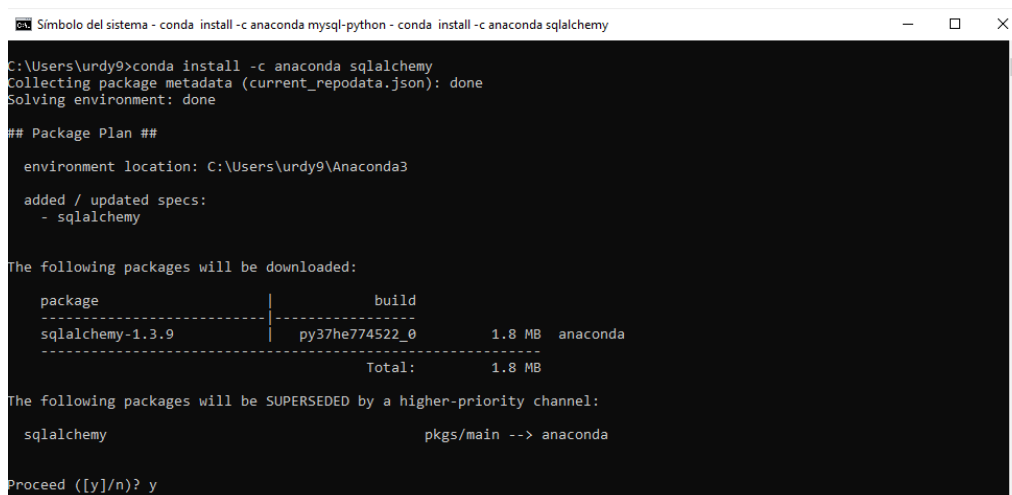
environment location: C:\Users\urdy9\Anaconda3

added / updated specs:
- cherry

The following packages will be downloaded:

package | build | size | channel
-----|-----|-----|-----
ca-certificates-2019.8.28 | 0 | 165 KB | anaconda
certifi-2019.9.11 | py37_0 | 155 KB | anaconda
```

Figura D.3: Instalación cherry.



```
Símbolo del sistema - conda install -c anaconda mysql-python - conda install -c anaconda sqlalchemy
C:\Users\urdy9>conda install -c anaconda sqlalchemy
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

environment location: C:\Users\urdy9\Anaconda3

added / updated specs:
- sqlalchemy

The following packages will be downloaded:

package | build | size | channel
-----|-----|-----|-----
sqlalchemy-1.3.9 | py37he774522_0 | 1.8 MB | anaconda
Total: 1.8 MB

The following packages will be SUPERSEDED by a higher-priority channel:
sqlalchemy pkgs/main --> anaconda

Proceed ([y]/n)? y
```

Figura D.4: Instalación SQL alchemy.

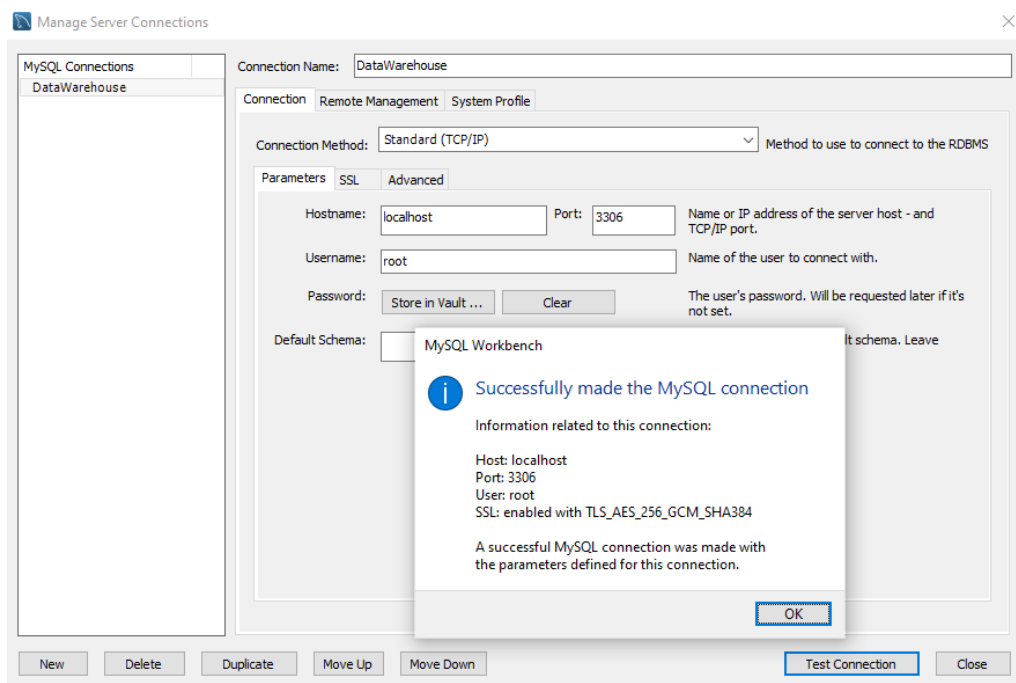


Figura D.5: Configuración conexión MySQL Workbench.

## MySQL Workbench

Para configurar la base de datos bastará con crear una conexión e importar la base de datos. Para crear la conexión deberemos abrir el programa y tras seleccionar el + rellenaremos los campos de la conexión [D.5](#).

Una vez hayamos configurado la conexión nos dirigiremos a server data import [D.6](#) y seleccionaremos el archivo con la base de datos a importar. Tras seleccionar el archivo a importar y el nombre del *schema* (deberemos dar a new y poner datawarehouse) [D.7](#), seleccionaremos la opción de *start import*.

## GitHub Desktop

Para la configuración de este servicio deberemos abrir la aplicación y en file, options, accounts iniciaremos sesión en nuestra cuenta de github, seleccionando el repositorio al que queremos añadir contenido.

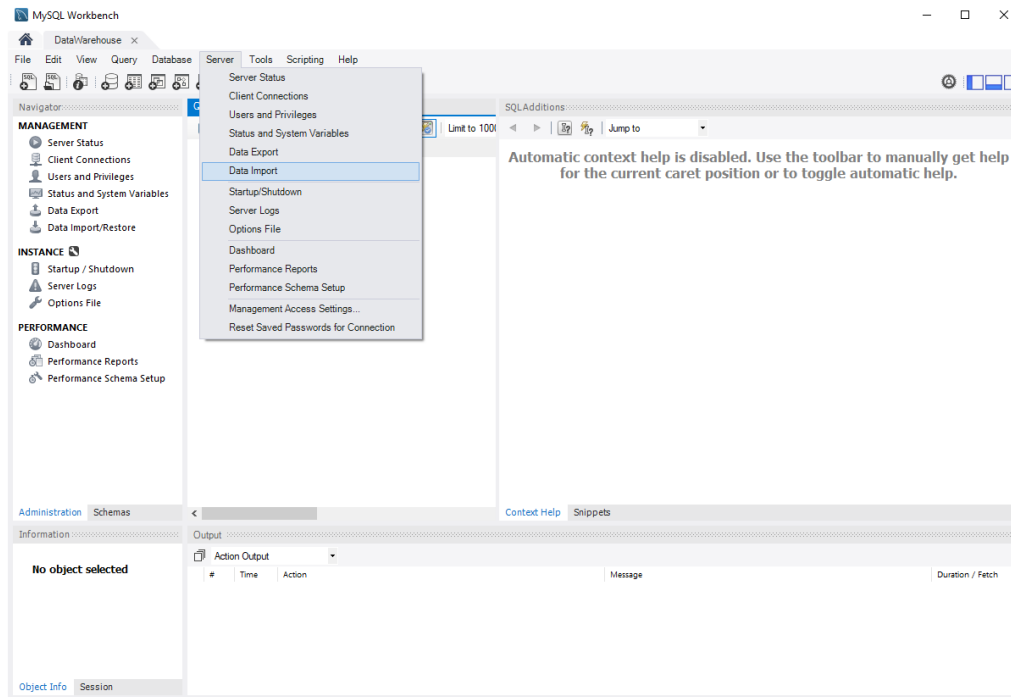


Figura D.6: Importar datos en MySQL.

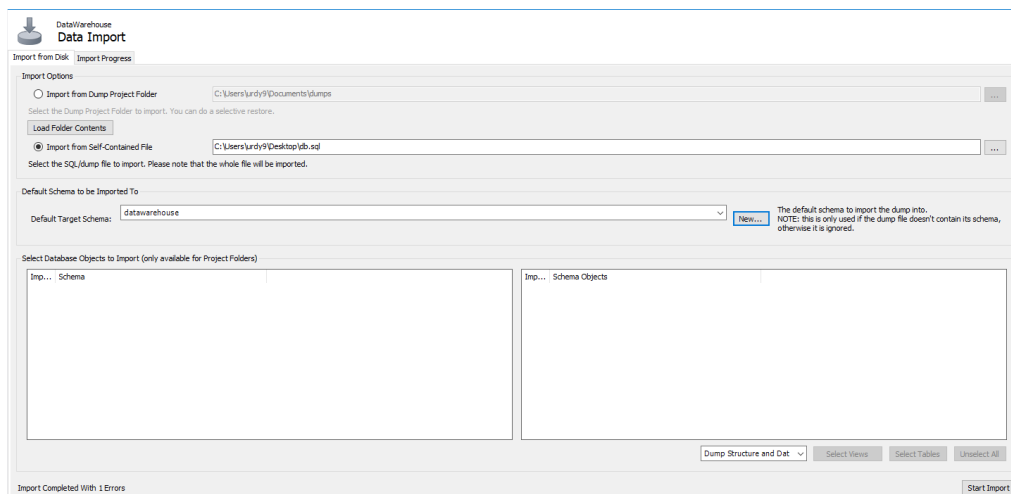


Figura D.7: Archivo a importar en MySQL.

## **Compilación y ejecución**

Para la compilación y ejecución mirar en el manual de usuario [E.3](#).



## *Apéndice E*

---

# Documentación de usuario

---

## E.1. Introducción

En este apartado se explicará los requisitos de la aplicación, como instalarla y cómo utilizarla correctamente.

## E.2. Requisitos de usuarios

Para el uso de la aplicación es necesario contar con los siguientes requisitos:

- Tener un ordenador con Windows instalado.
- Tener instalado **Microsoft Visual Studio** para la compatibilidad de los programas.
- Llevar a cabo la instalación de los programas que se indican en el manual del programador **D.3** en concreto MySQL workbench y Spyder.

## E.3. Instalación

Debido a la ausencia de un archivo ejecutable, para la instalación y ejecución del proyecto, es necesaria la compilación del mismo.

Para ello deberemos:

- Abrir MySQL Workbench metiendote en la base de datos (si nos pide contraseña, esta será la introducida en **D.3** 1234).

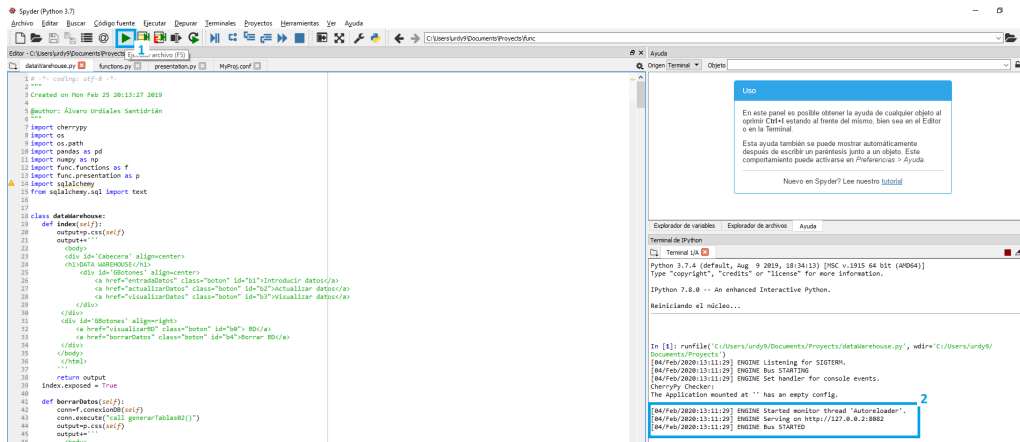


Figura E.1: Compilación y ejecución del proyecto.

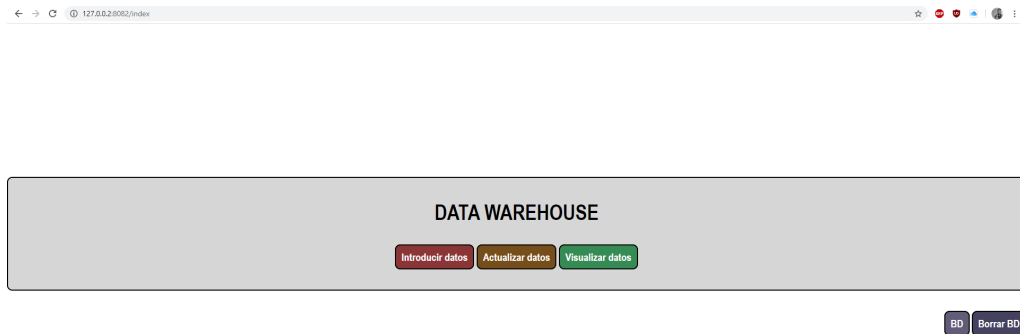


Figura E.2: Vista del proyecto en el navegador.

- Abrir Spyder y colocandonos en el archivo principal del proyecto pulsar F5 o el símbolo play verde que se encuentra en la parte superior del programa, de esta forma compiláramos y ejecutaríamos el programa [E.1](#) parte 1.

Una vez realizado, Spyder nos dirá una dirección web ([E.1](#) parte 2) donde introduciéndole en el navegador se encontrará el proyecto [E.2](#).



Figura E.3: Introducir datos nuevos en la aplicación.

## E.4. Manual del usuario

En esta sección explicaremos las diferentes funcionalidades de la aplicación.

### Introducir datos

En este apartado, se introducirán nuevos datos en la base de datos. Para ello deberemos:

1. Seleccionar la opción Introducir datos del menú principal.
2. Deberemos seleccionar el archivo a introducir teniendo en cuenta que se debe encontrar en el directorio raíz del proyecto [E.3](#).
3. Una vez seleccionado el archivo pulsaremos sobre la opción de enviar, mostrándonos en la siguiente pantalla uno de los siguientes mensajes:
  - Los datos se han leído correctamente: en caso de que se hayan leído e introducido en la base de datos.
  - ERROR: el archivo no se encuentra en el mismo directorio: en el caso de que el archivo no se encuentre en el directorio raíz.
  - ERROR: los datos de x no se han introducido: en caso de que haya habido algún error en la tabla x a la hora de introducirla en la base de datos.
  - ERROR: No has introducido ningún archivo: en caso de no haber seleccionado nada en la pantalla anterior.

## Actualizar datos

En este apartado, se actualizaran los datos existentes en la base de datos. Para ello deberemos:

1. Seleccionar la opción Actualizar datos del menú principal.
2. Deberemos seleccionar el archivo a introducir teniendo en cuenta que se debe encontrar en el directorio raíz del proyecto [E.4](#).
3. Una vez seleccionado el archivo pulsaremos sobre la opción de enviar, pudiéndonos mostrar una de las siguientes pantallas:
  - ERROR: el archivo no se encuentra en el mismo directorio: en el caso de que el archivo no se encuentre en el directorio raíz.
  - ERROR: No has introducido ningún archivo: en caso de no haber seleccionado nada en la pantalla anterior.
  - En caso de haber seleccionado un archivo en el directorio raíz, no nos saltará ningún error, dándonos paso a la siguiente pantalla [E.5](#).
4. En el caso de que todo haya ido bien, en la pantalla nos dará la opción de actualizar la base de datos sin más o poder ver los cambios existentes entre el archivo nuevo y la base de datos (opción Mostrar cambios).
  - Si seleccionamos la opción de mostrar cambios, nos saldrá por pantalla las tablas en las que se han encontrado cambios con los datos viejos (a borrar) y los datos nuevos (a introducir). Si se desea realizar la actualización deberemos seleccionar la opción Actualizar datos para ejecutar la actualización o Volver para cancelarlo todo.

## Visualizar datos

Esta opción se utilizará para acceder a la información de la base de datos, permitiéndonos obtener tablas con los datos organizados de forma que mejore la comprensión de los mismos.

Para poder acceder a esto deberemos seleccionar la opción Visualizar datos del menú de inicio. Una vez seleccionado, nos mostrará los diferentes

**ACTUALIZAR DATOS**

Seleccionar archivo Ningún archivo seleccionado Enviar

\*el archivo a seleccionar debe encontrarse en la misma ruta que el programa.

Volver

Figura E.4: Selección de archivo para la actualización de datos.

**ACTUALIZAR DATOS**

¿Que desea hacer?

Mostrar cambios Actualizar datos BD Volver

Figura E.5: Menú actualización de datos.

**VISUALIZAR DATOS**

SELECCIONA UNA OPCIÓN:

Asignaturas Grado Profesores en Asignaturas Profesores por grupo Horas Profesor

Volver

Figura E.6: Menú visualización de datos.

filtros que podemos aplicar a la base de datos para mostrar la información **E.6.**

Los diferentes filtros son:

- **Asignaturas grado:** Nos mostrará las diferentes asignaturas que hay en el grado ordenadas por curso y semestre.
- **Profesores en asignaturas:** Nos mostrará los diferentes profesores que hay en cada asignatura del grado ordenados por asignatura, curso y semestre.
- **Profesores por grupo:** Nos mostrará los diferentes profesores que hay en cada grupo de cada asignatura del grado ordenados por grupo,



The screenshot shows a web interface titled "VISUALIZAR ASIGNATURAS". Below the title, it says "SELECCIONA UN GRADO:". Underneath, the text "63 - GRADO EN INGENIERÍA INFORMÁTICA" is displayed. There is a text input field followed by a green button labeled "Enviar". In the bottom right corner, there is a dark blue button labeled "Volver".

Figura E.7: Menú selección de grado.

asignatura, curso y semestre.

- Horas profesor: Nos mostrará las diferentes horas que tienen de docencia asignadas cada uno de los profesores del grado dividido por semestres e indicando la suma de horas totales por año.

Una vez seleccionado el filtro deseado, nos saldrá una pantalla en la que nos listará los diferentes grados con su código asociado donde deberemos introducir el código del grado sobre el que queramos obtener la información [E.7](#).

Cuando hayamos introducido el código y pulsado sobre enviar, nos mostrará la tabla con la información requerida o un mensaje de error si se ha introducido un código de grado no válido.

---

## Bibliografía

---

- [1] Free Software Foundation. El sistema operativo gnu, 2019. [Internet; accedido 15-enero-2020].
- [2] MySQL. Mysql workbench, 2020. [Internet; accedido 18-enero-2020].
- [3] Spyder. Overview, 2018. [Internet; accedido 18-enero-2020].
- [4] Mark Webbink, 2005. [Internet; accedido 12-enero-2020].
- [5] Wikipedia. Especificación de requisitos de software, 2019. [Internet; accedido 5-enero-2020].
- [6] Wikipedia. Licencia de software, 2019. [Internet; accedido 12-enero-2020].
- [7] Wikipedia. Modelo–vista–presentador, 2020. [Internet; accedido 02-enero-2020].