



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**GII 20.09 Herramienta web
repositorios de TFGII
Documentación Técnica**



Presentado por David Renedo Gil
en Universidad de Burgos — 14 de diciembre
de 2022

Tutor: Álvaro Arnaiz González y Ana Serrano
Mamolar

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	6
Apéndice B Especificación de Requisitos	9
B.1. Introducción	9
B.2. Objetivos generales	9
B.3. Catalogo de requisitos	9
B.4. Especificación de requisitos	9
Apéndice C Especificación de diseño	11
C.1. Introducción	11
C.2. Diseño de datos	11
C.3. Diseño procedimental	11
C.4. Diseño arquitectónico	11
Apéndice D Documentación técnica de programación	13
D.1. Introducción	13
D.2. Estructura de directorios	13
D.3. Manual del programador	14

D.4. Compilación, instalación y ejecución del proyecto	19
D.5. Pruebas del sistema	30
Apéndice E Documentación de usuario	31
E.1. Introducción	31
E.2. Requisitos de usuarios	31
E.3. Instalación	31
E.4. Manual del usuario	31
Bibliografía	35

Índice de figuras

A.1. Gráfica Control chart- Sprint 1	3
A.2. Gráfica Control chart- Sprint 2	5
D.1. Descarga de JDK 11	14
D.2. Descarga JDK 11 Licencia	15
D.3. Seleccionar Eclipse	16
D.4. Eclipse marketplace	17
D.5. Plugin Vaadin	18
D.6. Copiar URL repositorio	19
D.7. Consola con Tomcat ejecutado	20
D.8. Gestor de Aplicaciones de Tomcat	20
D.9. Desplegar el archivo .war	21
D.10.Añadir servidor de Tomcat a Eclipse	22
D.11.Seleccionar carpeta contenedora de Tomcat	23
D.12.Añadir proyectos a servidor	24
D.13.Error tras desplegar el .war en el Gestor de Aplicaciones de Tomcat	25
D.14.Logs proporcionados por Tomcat	25
D.15.Logs proporcionados por Tomcat	26
D.16.Cambio de versión Dynamic Web Module	27
D.17.Página de acceso a GitHub estudiantes	28
D.18.Proceso de petición de GitHub for students	28
D.19.Aportar la información necesaria para la verificación	29
D.20.Pestaña final tras aplicar a la oferta	30
D.21.Créditos de nuestra cuenta	30
E.1. Preguntar al usuario si quiere actualizar la base de datos	32
E.2. Información sobre la EPS	32
E.3. Selección de parámetros	33

E.4. Gráfica final tras seleccionar los parámetros	33
E.5. Gráfica final tras seleccionar los parámetros	34

Índice de tablas

A.1. Licencias de las herramientas Software.	7
A.2. Planteamiento de horas iniciales.	8

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En esta sección se detallará la planificación que se ha realizado, el estudio de viabilidad tanto de la parte económica, como temporal y de la legal.

A.2. Planificación temporal

Se nombrarán y explicarán brevemente las tareas realizadas a lo largo del proyecto. Estas tareas se encuentran en el [repositorio del proyecto en Github](#).

Se añadirán gráficas para una mejor comprensión del tiempo que ha supuesto cada tarea en los ciclos (*Sprints*).

Sprint 0 - Puesta a punto (5/10/22 - 19/10/22)

Puesta a punto del proyecto. Se procederá a plantear las herramientas con las que se va a trabajar, búsqueda de alternativas y toma de contacto con las herramientas nuevas que se van a emplear.

A continuación se detallarán las tareas que se realizaron durante este primer Sprint:

- Añadir la extensión ZenHub al navegador. Desde el **Chrome Web Store** de Google Chrome se añadió la extensión **ZenHub for GitHub**.

- Clonar en repositorio en local. Para clonarlo se ha utilizado la herramienta **Github Desktop**. Mediante en enlace **HTTP** que proporciona *Github*.
- Documentación sobre Vaadin. Se procederá a estudiar el *framework* Vaadin con el que se va a trabajar. A través de la página oficial de **Vaadin** se realiza la instalación en nuestro entorno IDE **Eclipse** y el aprendizaje.
- Instalación JDK 11 o superior. Para utilizar la última versión de Vaadin se descargará el **openjdk 17**.
- Importación de un proyecto Vaadin de prueba a Eclipse. Para probar el correcto funcionamiento de Vaadin descargaremos e importaremos el proyecto de **prueba**.
- Clonación e imitación del repositorio en Eclipse. Trataremos de clonar e imitar el funcionamiento de la versión **anterior del proyecto** sobre la que trabajamos. Posteriormente se descargará también el **openjdk 11** para tratar de clonar el repositorio que estaba en la anterior versión del proyecto. También debemos instalar la herramienta **Tomcat**.
- Comienzo de la documentación. Para ello hemos instalado las herramientas TexStudio y MikTex como se indica en **plantillaLatex** y se ha buscado información para iniciar la documentación.
- Actualización del README.md. Se modificó el README.md del proyecto para que refleje los cambios respecto a la versión anterior.
- Búsqueda de trabajos relacionados con la gestión de TFG/TFM. Se realizó una investigación con el fin de encontrar proyectos similares a la aplicación web, es decir, que consistan en la gestión de trabajos de fin de grado o similares. Los proyectos encontrados serán explicados en el apartado **Trabajos relacionados** de la memoria.

Sprint 1 - (19/10/22 - 9/11/22)

Se procederá a estudiar el código del repositorio y a documentar el anexo.

A continuación se detallarán las tareas que se realizaron durante este primer Sprint:

- Comienzo de la documentación del anexo. Comenzamos en este Sprint a realizar esta documentación desde TexStudio.

- Estudio del código de todos los paquetes de la carpeta src. Tanto persistence, como util, ui, security y webService.
- Se procede a buscar el error que salta al intentar ejecutar el código en local.

Se puede ver el trascurso de estas tareas en la ilustración A.1.

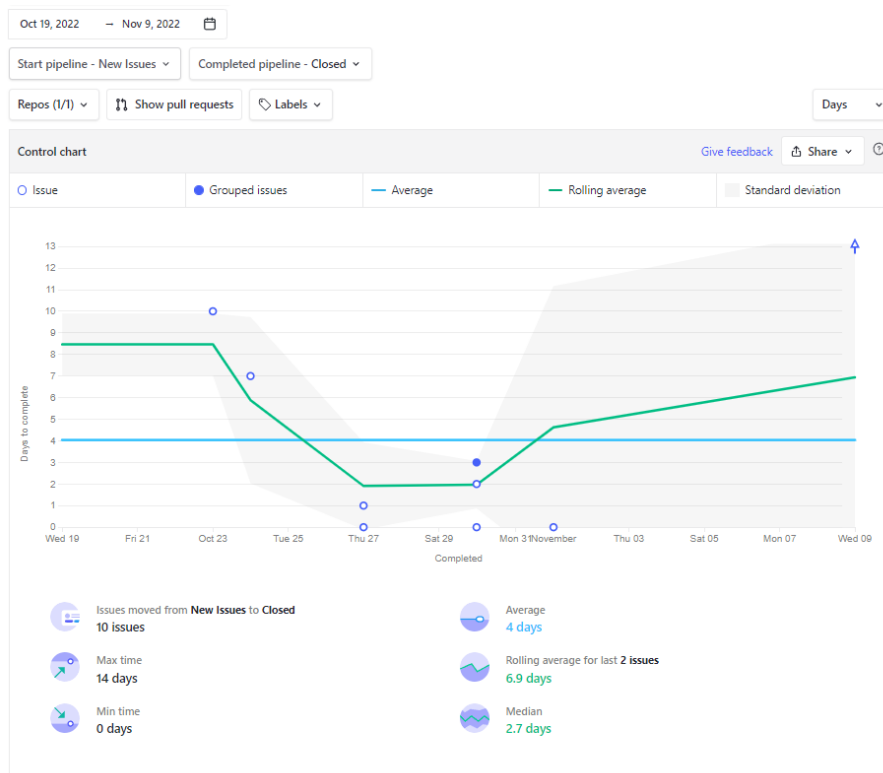


Figura A.1: Gráfica Control chart- Sprint 1

Sprint 2 - Comienzo de la programación (10/11/22 - 23/11/22)

En este sprint se comienza a programar y añadir código principalmente arreglando bugs que existían en la versión anterior. También se investiga sobre una alternativa al uso de Heroku que ahora es de pago.

A continuación se detallarán las tareas que se realizaron durante este segundo Sprint:

- Eliminación de la distinción entre mayúsculas y minúscula en los filtros. Anteriormente se tenía que introducir el nombre exacto en una columna para que se aplicase bien el filtro, ahora no existe esa distinción.
- Url del apartado *Documentos* era errónea y se ha sustituido por la correcta.
- Actualización apartado *información*. Se ha actualizado la información respecto a los tutores y la última versión.
- Investigar sobre el webscrapping. En un futuro se deberá realizar un webscrapping con la página de investigación de la ubu, por lo que se ha estudiado en qué consiste y posibles implementaciones.
- Investigación estadística errónea. La información sobre las columnas *Nota*, *TotalDias* y *Repositorio* estaba mal implementada en el archivo `BaseDeDatosTFGTFM.xls` y se ha cambiado a un formato adecuado.
- Se actualiza la memoria y el anexo correspondiente al anterior Sprint.
- Elección de una alternativa a Heroku. Heroku pasa a ser de pago el 28 de noviembre de 2022, por lo que se han buscado diferentes alternativas gratuitas como la versión de [Heroku para estudiantes](#), una colaboración entre Heroku y [GitHub for Students](#) o [Northflank](#).
- Se realizan diferentes pruebas en las plataformas para decidir cual utilizar, y finalmente se optará por usar Heroku for Students, tras desplegar el proyecto en Heroku con éxito y que al tratar de importarlo a Northflank nos indica que debemos aportar una tasa.
- Búsqueda de librerías o APIs para realizar el webscraping en nuestro proyecto. Se analizan algunas librerías como [Jsoup](#), [HTMLUnit](#) o [Jaunt](#) y APIs como [Octoparse](#).
- Pruebas de webscraping en un entorno local para determinar cual utilizar. Se realizan una serie de pruebas (que encontramos en el apartado de Pruebas de la Documentación) y finalmente se consigue obtener el resultado que queremos mediante JSoup, por lo que será nuestra elección.

Se puede ver el trascurso de estas tareas en la ilustración [A.2](#).

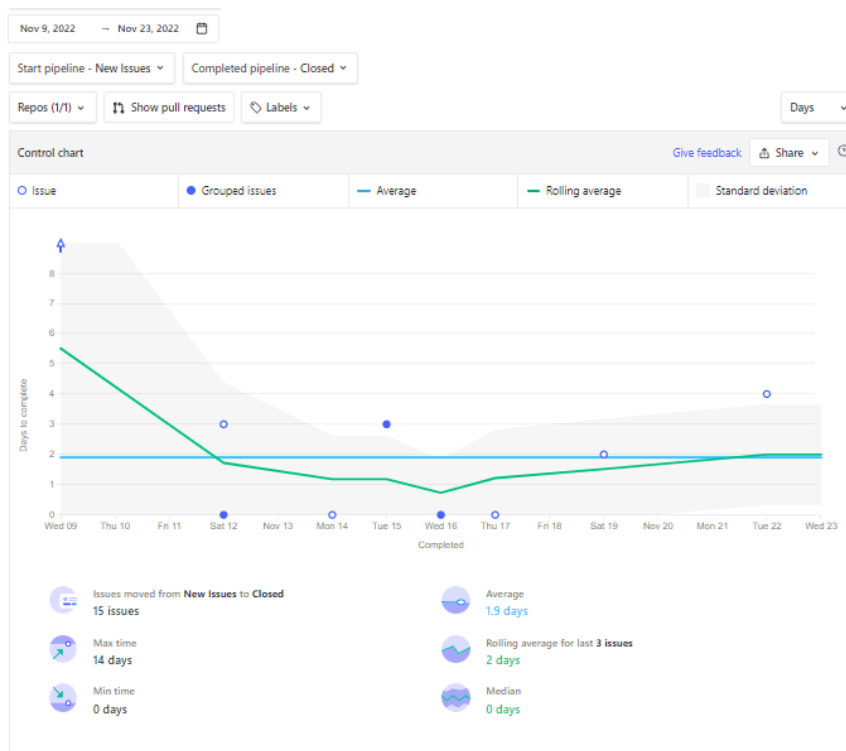


Figura A.2: Gráfica Control chart- Sprint 2

Sprint 3 - Implementación nuevas pantallas (23/11/22 - 14/12/22)

En este sprint se implementará al proyecto el proceso de *webscraping* llevado a cabo en el sprint anterior, también se crearán dos nuevas pantallas.

A continuación se detallarán las tareas que se realizaron durante este segundo Sprint:

- Guardar los datos del *webscraping* en un archivo csv/xls. Se programa un código que permite guardar los resultados sacados mediante el *webscraping* a los ficheros `BaseDeDatosTFGTfM.xls` y `N4 Profesores.csv`.
- Creación del *mock-up* pantalla de creación de informe. Mediante **Pencil** crearemos una vista inicial de lo que queremos que sea nuestra pantalla. Esta pantalla se utilizará para crear un informe de un determinado área a elegir por el usuario y guardará los datos en un archivo `.xls`.

- Creación del *mock-up* pantalla de estadísticas del profesorado. Creada también mediante **Pencil**. Esta pantalla se utilizará para visualizar los históricos de los profesores que queramos de la *EPS*, dependiendo de los departamentos y áreas que se indiquen.
- Implementación *webscrap* en nuestro proyecto. Se ha introducido esta función en nuestro proyecto.
- Implementación de la pantalla Generar Informes. Se comienza y se termina de programar esta nueva pantalla. No se han creado más *issues* si no que se iba comentando en esta *issue* los problemas y el continuo desarrollo de la pantalla, así como las dudas planteadas.
- Implementación de la pantalla Estadísticas del Profesorado. Se comienza y se termina de programar esta nueva pantalla.
- Corrección de la memoria y anexos. Se corrige los fallos expuestos tras el *feedback* del tutor Álar Arnaiz de los ficheros latex de memoria y anexo.

Sprint 4 -

A.3. Estudio de viabilidad

En este apartado se detallan los costes que llevaría realizar este proyecto de forma real. Se considerarán los costes de recursos humanos, el material empleado y el *Software* usado.

Viabilidad económica

Los recursos utilizados para la realización de este proyecto son los siguientes:

- **Lenovo Legion Y540** Coste aproximado: 1200€.
- **Eclipse IDE** como entorno de desarrollo del código. Coste: gratuito.
- **TexStudio** para realizar la documentación. Coste: gratuito.
- **Github Desktop**: como herramienta para actualizar el directorio de github. Coste: gratuito.
- **Heroku**: como herramienta de despliegue del proyecto en la nube. Coste: 5 €/mes aproximado.

Software	Licencia
Vaadin	Apache License 2.0
Spring Boot Maven Plugin	Apache License 2.0
JUnit	Eclipse Public License 1.0
CsvJdbc	LGPLv2
Codoid Fillo	Apache License, Version 2.0

Tabla A.1: Licencias de las herramientas Software.

- **Maven:**. Coste: gratuito.
- **Tomcat:**. Coste: gratuito.
- **Tiempo empleado:** aproximadamente 250 horas, que con el salario medio español de un programador (14,43 €/hora) es 3607,50 €. Fuente [Talent.com](#)

Viabilidad legal

Se detallaran las licencias *Software* de cada dependencia que se ha utilizado en el proyecto. En el proyecto se ha usado la licencia MIT que permite la libre distribución del *software*.

También existe una cuestión de legalidad a la hora de hacer *webscraping*, ya que no siempre es legal realizar este tipo de acciones sobre algunas páginas web, sobretodo si no tenemos los permisos necesarios. En nuestro caso no aplica ya que lo realizamos sobre una *web* interna, por lo que no existe ningún conflicto a la hora de obtener la información.

Planificación temporal

En esta sección se mostrará una ideal inicial de como iba a ser el reparto de horas de las diferentes tareas durante el desarrollo del TFG. La tabla [A.3](#) muestra una predicción de la división del proyecto.

TAREA	TIEMPO
Instalación y configuración de software y hardware	6 h
Programación	140 h
Documentación memoria y anexos	60 h
Estudio de las herramientas utilizadas y alternativas	10 h
Preparación de la presentación	5 h

Tabla A.2: Planteamiento de horas iniciales.

Apéndice B

Especificación de Requisitos

B.1. Introducción

La especificación de requisitos hace referencia a los requerimientos que debe cumplir el software para satisfacer las necesidades del cliente. Debe incluir la suficiente cantidad de detalles para permitir a los desarrolladores software diseñar el sistema. Solo se incluirán los requisitos realizados en esta mejora.

B.2. Objetivos generales

El objetivo general del proyecto es continuar con el desarrollo y la mejora de la aplicación web respecto a la versión anterior, centrándose en los siguientes puntos:

B.3. Catalogo de requisitos

B.4. Especificación de requisitos

Apéndice C

Especificación de diseño

C.1. Introducción

En este anexo se detallarán los aspectos referentes al diseño de la aplicación en esta mejora de la aplicación.

C.2. Diseño de datos

C.3. Diseño procedimental

C.4. Diseño arquitectónico

Apéndice *D*

Documentación técnica de programación

D.1. Introducción

En esta sección se van a detallar los diferentes procesos de instalación de las herramientas que se han utilizado durante el proyecto. También se especificará la estructura del proyecto, instalación de dependencias, la compilación, la ejecución del proyecto y el despliegue en Heroku.

D.2. Estructura de directorios

Se enumerarán y describirán brevemente los directorios del proyecto. Se puede encontrar el código fuente en el repositorio de Github denominado “Gestor-TFG-2022”.

- `/`: directorio raíz donde se ubican el README, Maven.
- `/.github/workflows` los archivos de *workflow* o flujo de trabajo, tanto para la Integración continua del proyecto en GitHub como para el análisis de la calidad del código en [SonarCloud](#).
- `/Documentacion` material de documentación del proyecto y prueba empleadas.
 - `/Documentacion/LaTeX` ficheros para generar la memoria y los anexos realizados en *TexStudio*.

- /Documentacion/Pruebas aplicaciones prototipo para comenzar el aprendizaje con **Vaadin** y pruebas realizadas con diferentes librerías durante el webscraping.

D.3. Manual del programador

A continuación se detallará el proceso de instalación de los programas necesarios para el desarrollo de la aplicación.

Instalación de Java

Actualmente se sigue ejecutando con la versión de Java 11. A pesar de que se necesitará actualizar cuando migremos a la versión 23 de vaadin.

Para ello se debe descargar la [página de descargas de Oracle Java SE 11.0](#) y descargar la versión de JDK 11, correspondiente con el sistema operativo que se posea y su arquitectura, ya sea de 64 o 32 bits. Ver imagen [D.1](#).

Tras escoger la versión según el SO, se leerán y aceptarán las licencias de uso de Oracle [D.2](#), y se hará *click* en descargar.












This software is licensed under the Oracle Technology Network License Agreement for Oracle Java SE		
Product / File Description	File Size	Download
Linux ARM 64 Debian Package	145.64 MB	 jdk-11.0.10_linux-aarch64_bin.deb
Linux ARM 64 RPM Package	152.22 MB	 jdk-11.0.10_linux-aarch64_bin.rpm
Linux ARM 64 Compressed Archive	169.37 MB	 jdk-11.0.10_linux-aarch64_bin.tar.gz
Linux x64 Debian Package	149.39 MB	 jdk-11.0.10_linux-x64_bin.deb
Linux x64 RPM Package	156.12 MB	 jdk-11.0.10_linux-x64_bin.rpm
Linux x64 Compressed Archive	173.31 MB	 jdk-11.0.10_linux-x64_bin.tar.gz
macOS Installer	167.51 MB	 jdk-11.0.10_osx-x64_bin.dmg
macOS Compressed Archive	167.84 MB	 jdk-11.0.10_osx-x64_bin.tar.gz
Solaris SPARC Compressed Archive	184.82 MB	 jdk-11.0.10_solaris-sparcv9_bin.tar.gz
Windows x64 Installer	152.32 MB	 jdk-11.0.10_windows-x64_bin.exe
Windows x64 Compressed Archive	171.67 MB	 jdk-11.0.10_windows-x64_bin.zip

Figura D.1: Descarga de JDK 11

También se deberá cambiar la variable de entorno de Java del sistema.

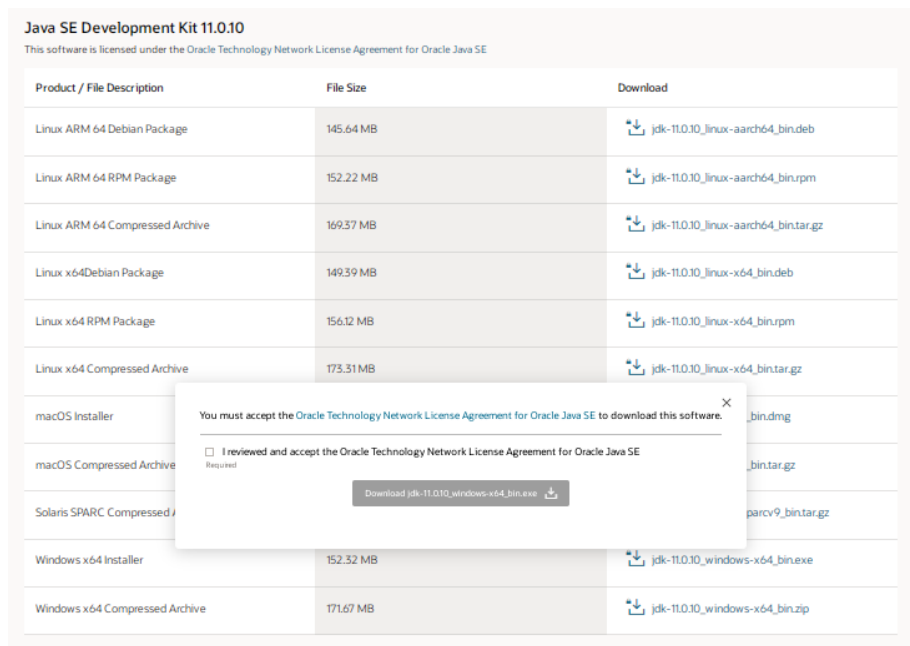


Figura D.2: Descarga JDK 11 Licencia

Instalación de Eclipse

A continuación se instalará un entorno de desarrollo integrado (IDE) para Java, en este caso se ha utilizado **Eclipse IDE for Enterprise Java Developers** en la versión 2021-12.

Para descargar el IDE se accederá a la [página de descargas de Eclipse](#) y descargar la opción correspondiente a nuestro sistema operativo del **Eclipse Installer 2021-12 R**.

En el caso de los sistemas operativos Windows se descargará un archivo ejecutable que se deberá ejecutar como administrador. Una vez ejecutado se deberá seleccionar la opción “**Eclipse IDE for Enterprise Java Developers**” [D.3](#).

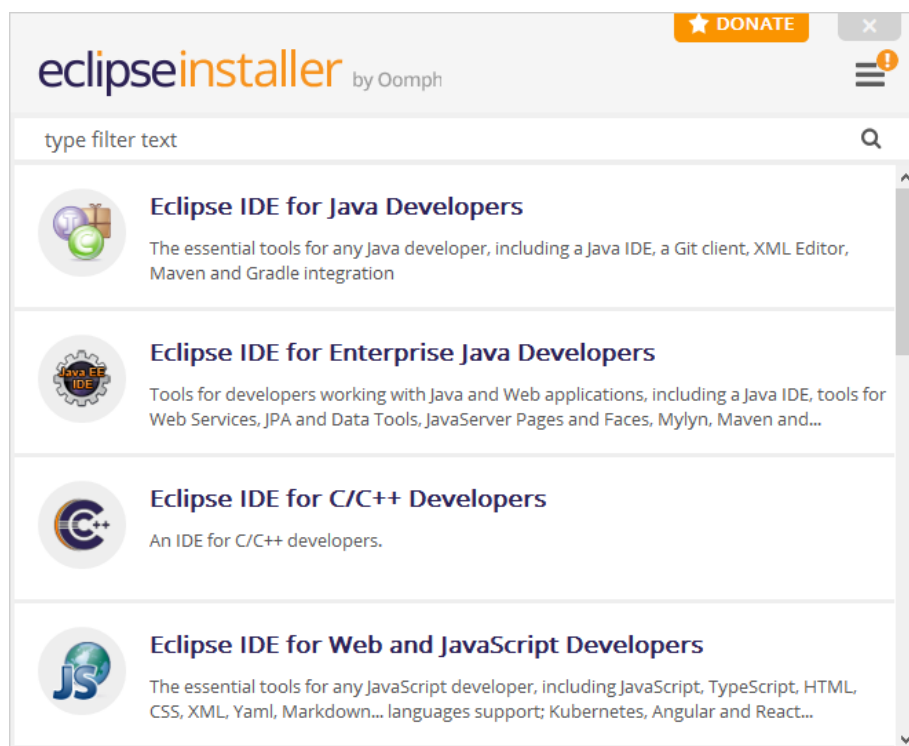


Figura D.3: Seleccionar Eclipse

Por último seleccionaremos el JDK (11) que vayamos a utilizar y la carpeta donde queremos instalar nuestro IDE.

Instalación del *plugin de Vaadin* para Eclipse

Una vez se haya instalado Eclipse, se procederá a añadir el plugin de Vaadin para Eclipse. Esto se realizará mediante el **Eclipse Marketplace de Eclipse D.4**, el cual se encuentra en la opción de “**Help/Eclipse Marketplace...**” de la barra de herramientas.

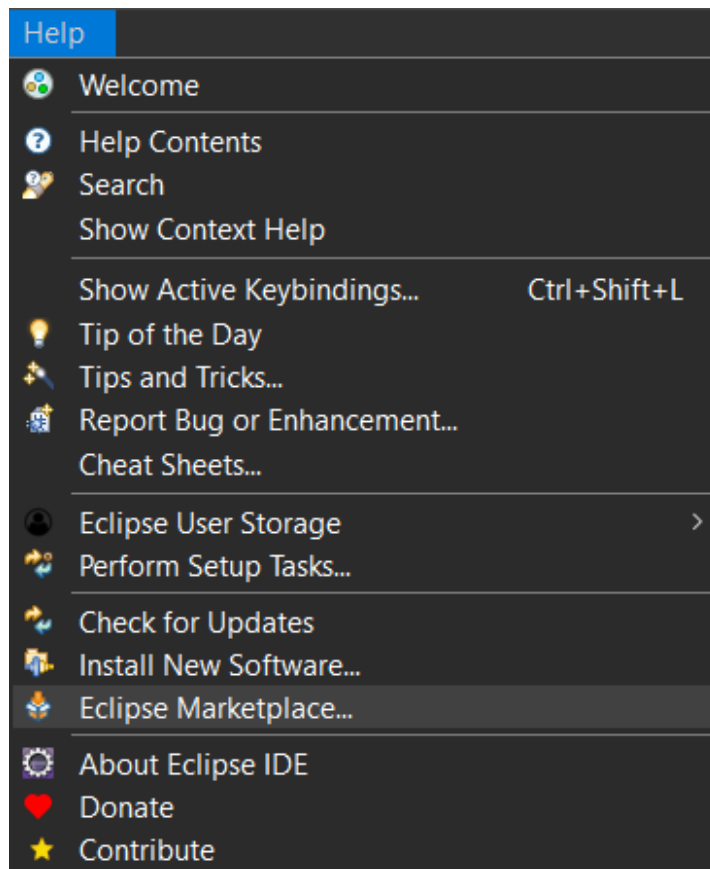


Figura D.4: Eclipse marketplace

Una vez en el Eclipse Marketplace, se buscará “**Vaadin**” y se pulsará “**Go**”. Tras salir el plugin “***Vaadin Plugin for Eclipse***”, se dará a “**Install**” y comenzará la instalación del plugin [D.5](#). En la imagen ya se muestra una vez instalado.

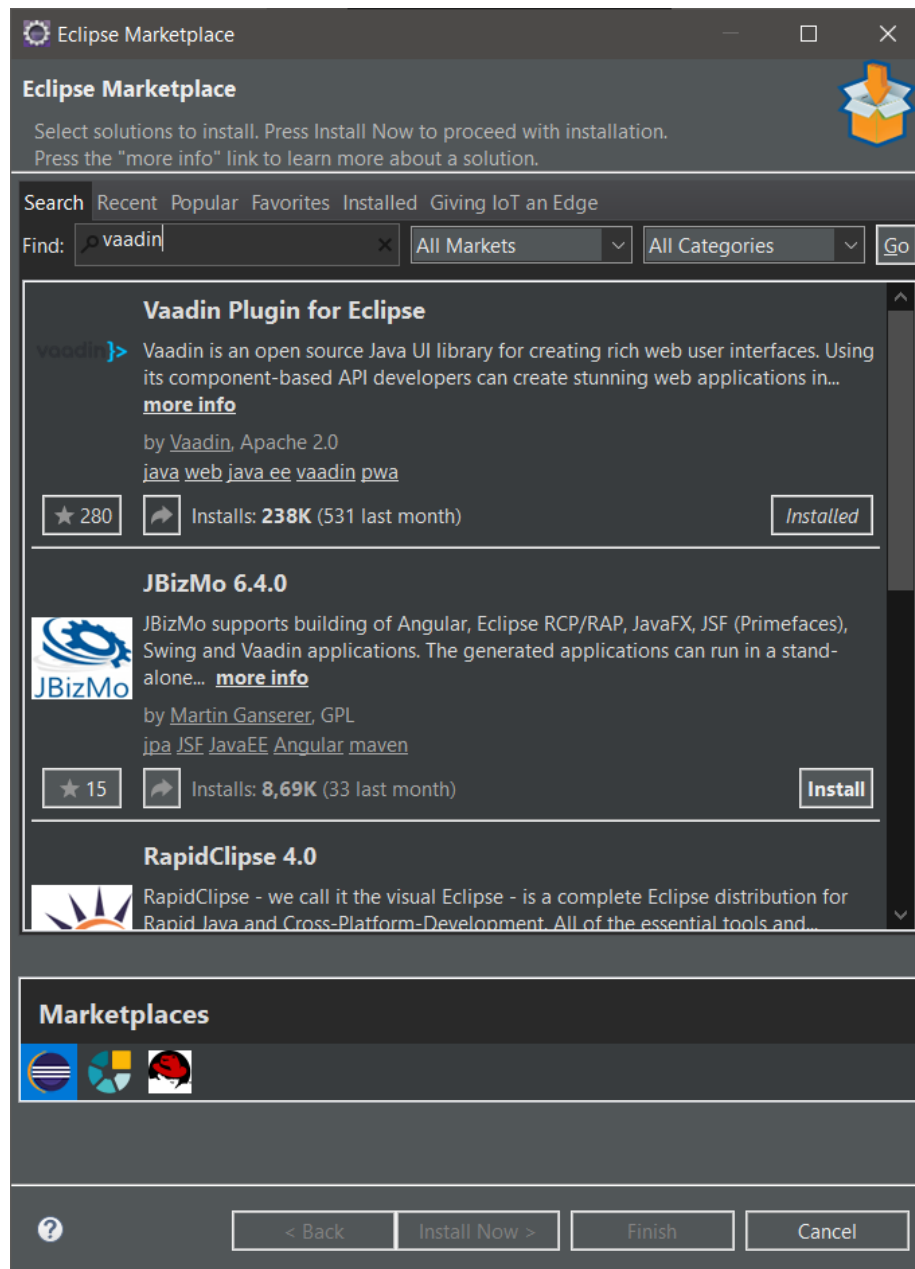


Figura D.5: Plugin Vaadin

D.4. Compilación, instalación y ejecución del proyecto

Se explicará como compilar, instalar y ejecutar el proyecto. En el caso de la ejecución, se detallará como hacerlo desde un terminal y mediante Eclipse (IDE).

Descarga del repositorio

El código fuente se encuentra en el **repositorio del proyecto** en GitHub. Para descargarlo se deberá hacer click en “**Code**” y copiar la URL que aparece en el apartado de “**HTTP**”. Con esta URL deberemos ir al “**GitHub Desktop**” y clonar el repositorio **D.6**.

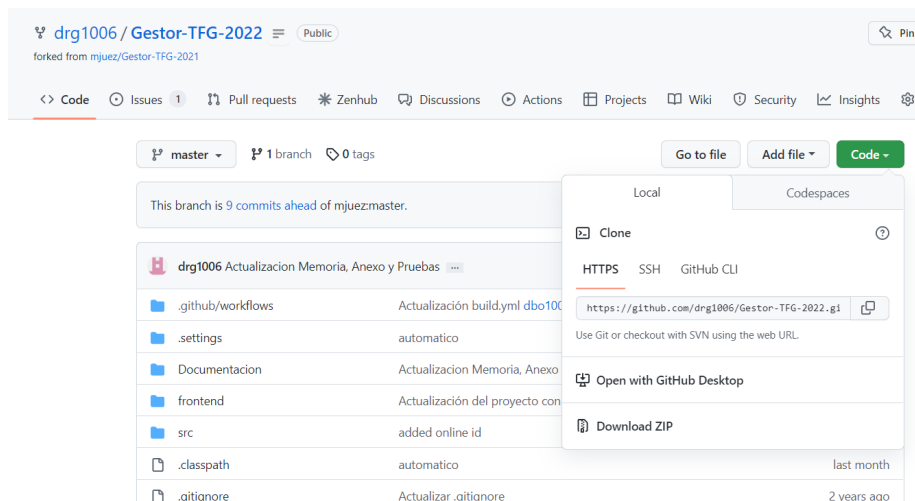


Figura D.6: Copiar URL repositorio

Si se desea tener código en local se deberá descargar el zip “**Download ZIP**” en la opción “**Code**” anteriormente mencionada. Una vez descargado el fichero se descomprimirá y abrirá con Eclipse.

Compilación del proyecto

Para compilar el proyecto en local desde terminal se usará:

- Limpiar las dependencias: `mvn clean`.
- Instalar dependencias y compilar: `mvn install`.

- Instalar en modo producción (para desplegar): `mvn package -Pproduction`.
- Ejecutar test: `mvn test`.

Ejecución del proyecto desde local

Para la ejecución del proyecto en local desde terminal se usará:

- Entrar en la terminal que utilizemos.
- Acceder a la carpeta donde tenemos nuestro servidor tomcat instalado y entrar en la carpeta `/bin`.
- Ejecutar nuestro servidor local mediante **startup** D.7.
- Entrar en el nuestro navegador en la dirección **localhost:8080**.
- Pulsar en la opción Manage App D.8.
- Iniciamos sesión como manager-gui. (Indicado en el archivo `/conf/tomcat-users.xml`).
- Llegaremos a la pantalla D.9 y seleccionaremos el archivo `.war` que hemos creado al compilar nuestro proyecto con “**mvn package -Pproduction**”.

```
C:\Programas\apache-tomcat-9.0.68\bin>startup
Using CATALINA_BASE:   "C:\Programas\apache-tomcat-9.0.68"
Using CATALINA_HOME:   "C:\Programas\apache-tomcat-9.0.68"
Using CATALINA_TMPDIR: "C:\Programas\apache-tomcat-9.0.68\temp"
Using JRE_HOME:        "C:\Program Files\Java\jdk-11.0.16"
Using CLASSPATH:       "C:\Programas\apache-tomcat-9.0.68\bin\bootstrap.jar;C:\Programas\apache-tomcat-9.0.68\bin\tomcat-juli.jar"
Using CATALINA_OPTS:   ""
```

Figura D.7: Consola con Tomcat ejecutado

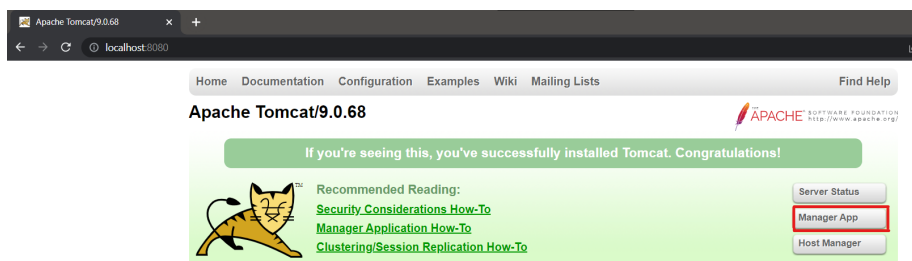


Figura D.8: Gestor de Aplicaciones de Tomcat

The screenshot shows the Tomcat Manager web interface in a browser window. The address bar shows the URL: `localhost:8080/manager/html/stop?path=/sistinf-0.8&org.apache.catalina.filters.CSRF_NONCE=966D0393E8A7A8669DF26022D85F919`. The interface is divided into several sections:

- Table of Applications:** A table with columns for application name, path, description, status, and actions. The rows are:

Application	Path	Description	Status	Actions
host-manager	Ninguno especificado	Tomcat Host Manager Application	true	Expirar sesiones sin trabajar 30 minutos
manager	Ninguno especificado	Tomcat Manager Application	true	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar 30 minutos
sistinf-0.7	Ninguno especificado		false	Arrancar Parar Recargar Replegar
sistinf-0.8	Ninguno especificado		false	Arrancar Parar Recargar Replegar
- Desplegar (Deploy):** A section with a title bar and a form for deploying a WAR file or directory. It includes fields for:
 - Trayectoria de Contexto (opcional):
 - Version (for parallel deployment):
 - URL de archivo de Configuración XML:
 - URL de WAR o Directorio:
 A "Desplegar" button is at the bottom.
- Archivo WAR a desplegar (WAR File to Deploy):** A section with a title bar and a form for selecting a WAR file. It includes a label "Seleccione archivo WAR a cargar" and a button "Seleccionar archivo". A "Desplegar" button is at the bottom.
- Configuration:** A section with a title bar and a form for configuration. It includes a label "Re-read TLS configuration files" and a field for "TLS host name (optional)". A "Re-read" button is at the bottom.

Figura D.9: Desplegar el archivo .war

Ejecución del proyecto desde Eclipse IDE

Para la ejecución del proyecto en local desde Eclipse primero debemos importar como proyecto Maven, con el **pom.xml**. Utilizaremos también un servidor local de **Apache Tomcat**, en concreto, la versión 9. Se puede descargar en [la página oficial de Apache Tomcat](#).

Una vez descargado y descomprimido, se creará un servicio de Tomcat, ver imagen [D.10](#), con la ruta donde se tiene descargado Tomcat y se le dará un nombre, ver imagen [D.11](#). Por último, se añadirá el proyecto principal “sistinf”, ver imagen [D.12](#).

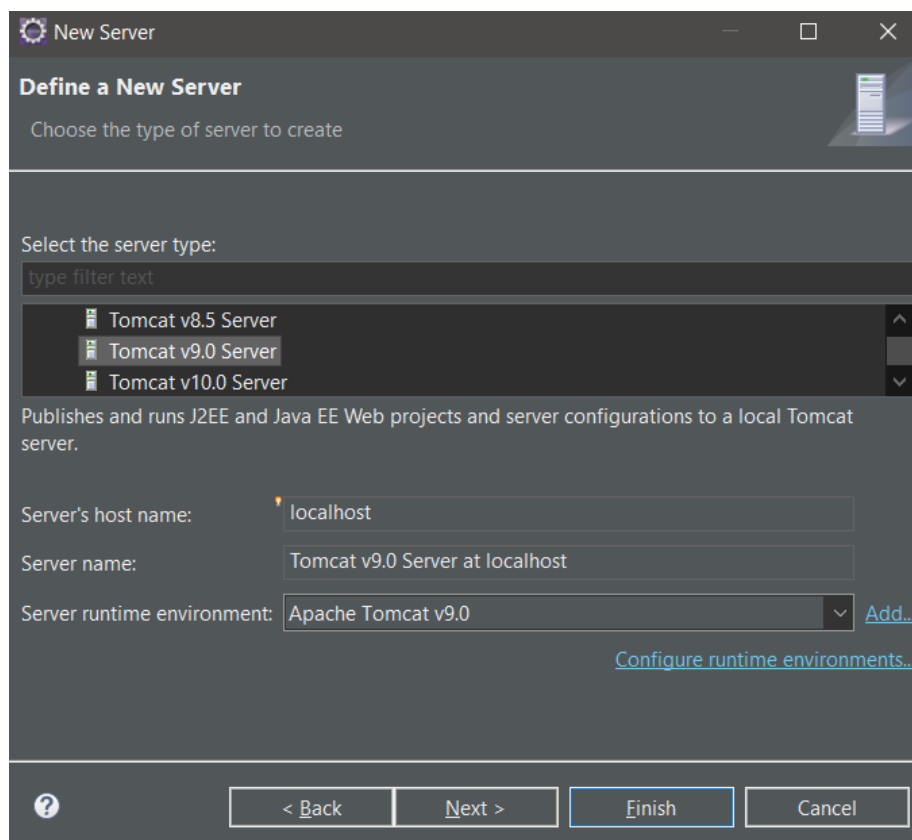


Figura D.10: Añadir servidor de Tomcat a Eclipse

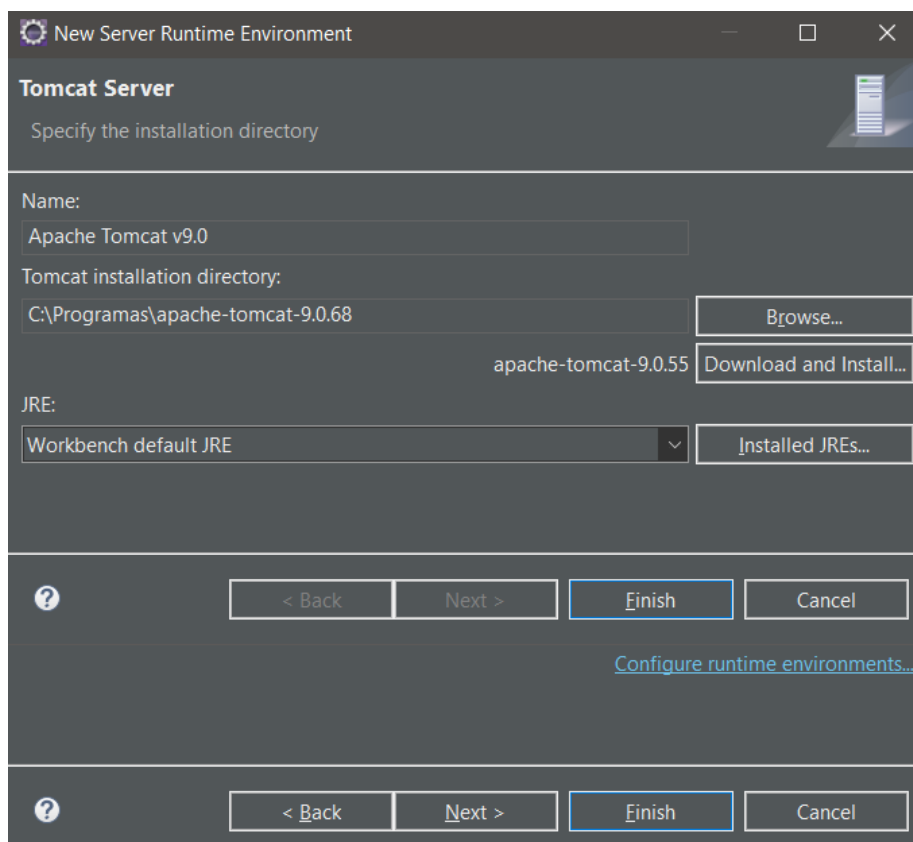


Figura D.11: Seleccionar carpeta contenedora de Tomcat

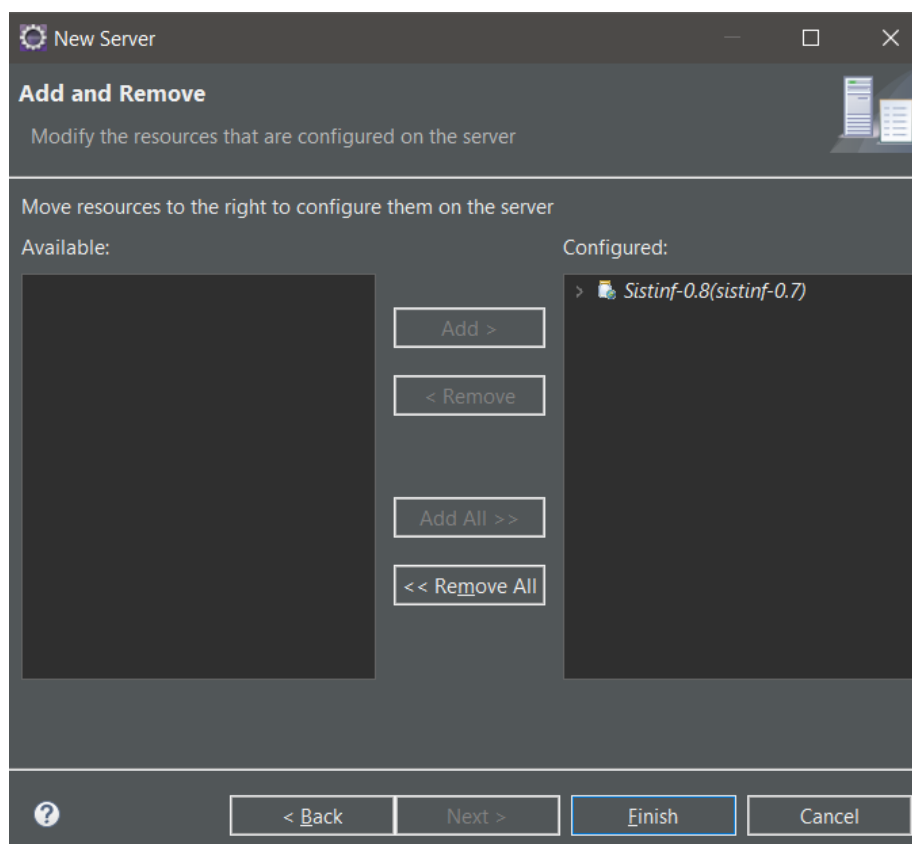


Figura D.12: Añadir proyectos a servidor

Para ejecutarlo desde Eclipse debemos también seguir todos los pasos de compilación anteriormente mencionados.

Una vez tengamos compilado nuestro código debemos ejecutarlo (click derecho en el proyecto → **Run As** → **Run on Server**).

Si no aparece la vista de los servicios se puede añadir desde la barra de herramientas → **Window** → **Show View**. Para configurar la ruta donde se ejecuta la aplicación, por defecto en **localhost:8080/** o en ciertos casos **localhost:8080/sistinf**.

Problemas a la hora de ejecutar el proyecto

A la hora de ejecutar el proyecto anterior surgieron una serie de problemas tanto para la ejecución por terminal como desde Eclipse.

Cuando quise ejecutarlo mediante la terminal desplegando el archivo .war generado tras compilar me surgió el siguiente error **D.13**

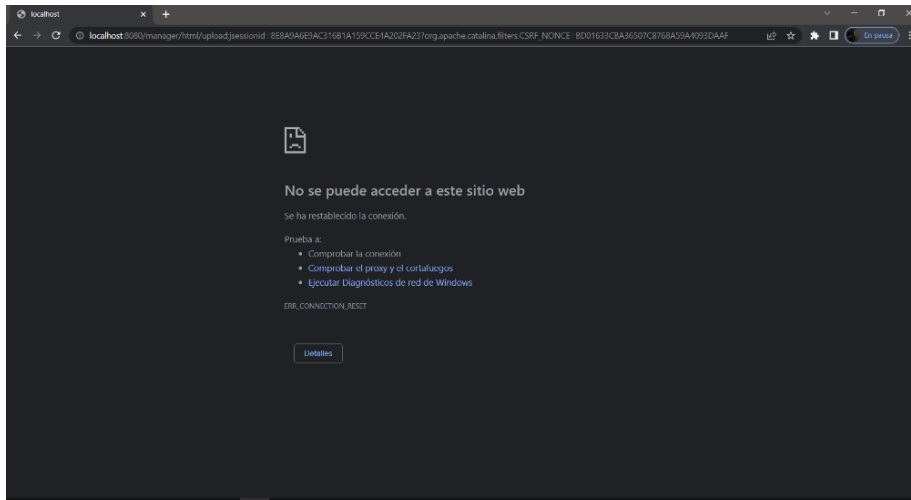


Figura D.13: Error tras desplegar el .war en el Gestor de Aplicaciones de Tomcat

Tras buscar información sobre el posible error, se descubre en los logs que proporciona tomcat lo siguiente D.14. En el que se informa que se intenta ejecutar un proyecto con un tamaño mayor al que tenemos configurado en tomcat.

```
08-Nov-2022 19:57:41.644 INFO [http-nio-8080-exec-3] org.apache.catalina.core.ApplicationContext.log HTTP/Manager: Init: Associated with Deployer 'CatalinaTypeDeployer_host-localhost'
08-Nov-2022 19:57:41.644 INFO [http-nio-8080-exec-2] org.apache.catalina.core.ApplicationContext.log HTTP/Manager: Init: Global resources are available
08-Nov-2022 19:57:41.651 INFO [http-nio-8080-exec-2] org.apache.catalina.core.ApplicationContext.log HTTP/Manager: Init: Listing contexts for virtual host 'localhost'
08-Nov-2022 19:58:02.480 SEVERE [http-nio-8080-exec-4] org.apache.catalina.core.ApplicationContext.log HTTP/Manager: FALLO - Falló carga de Despliegue, excepción: [org.apache.tomcat.util.http.fileupload.impl.Size
Java.lang.IllegalStateException: org.apache.tomcat.util.http.fileupload.impl.Size:161ExceededException: the request was rejected because its size (72771942) exceeds the configured maximum (52428800)
at org.apache.catalina.connector.Request.parseParts(Request.java:2974)
at org.apache.catalina.connector.Request.getParameter(Request.java:1263)
at org.apache.catalina.connector.Request.getParameter(Request.java:1141)
at org.apache.catalina.connector.RequestFacade.getParameter(RequestFacade.java:381)
at org.apache.catalina.filters.CsrfPreventionFilter.doFilter(CsrfPreventionFilter.java:127)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:189)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
at org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:189)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
at org.apache.catalina.filters.HttpHeaderSecurityFilter.doFilter(HttpHeaderSecurityFilter.java:126)
at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:189)
at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:162)
at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:197)
at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:197)
at org.apache.catalina.authenticator.AuthenticatorBase.invoke(AuthenticatorBase.java:688)
at org.apache.catalina.valves.RequestFilterValve.process(RequestFilterValve.java:178)
at org.apache.catalina.valves.RemoteAddrValve.invoke(RemoteAddrValve.java:86)
at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:135)
at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:92)
at org.apache.catalina.valves.AbstractAccessLogValve.invoke(AbstractAccessLogValve.java:687)
at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:78)
at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:368)
at org.apache.coyote.http11.Http11Processor.service(Http11Processor.java:398)
at org.apache.coyote.AbstractProcessorLight.process(AbstractProcessorLight.java:85)
at org.apache.coyote.AbstractProtocol$ConnectionHandler.process(AbstractProtocol.java:893)
at org.apache.tomcat.util.net.NioEndpoint$SocketProcessor.doRun(NioEndpoint.java:1789)
at org.apache.tomcat.util.net.SocketProcessorBase.run(SocketProcessorBase.java:49)
at org.apache.tomcat.util.threads.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1191)
at org.apache.tomcat.util.threads.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:658)
at org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)
at java.base/java.lang.Thread.run(Thread.java:834)
Caused by: org.apache.tomcat.util.http.fileupload.impl.Size:161ExceededException: the request was rejected because its size (72771942) exceeds the configured maximum (52428800)
at org.apache.tomcat.util.http.fileupload.impl.FileItemIteratorImpl.init(FileItemIteratorImpl.java:161)
at org.apache.tomcat.util.http.fileupload.impl.FileItemIteratorImpl.getInputStream(FileItemIteratorImpl.java:205)
at org.apache.tomcat.util.http.fileupload.impl.FileItemIteratorImpl.findNextItem(FileItemIteratorImpl.java:224)
at org.apache.tomcat.util.http.fileupload.impl.FileItemIteratorImpl.cinit(FileItemIteratorImpl.java:142)
at org.apache.tomcat.util.http.fileupload.impl.FileUploadBase.getItemIterator(FileUploadBase.java:252)
at org.apache.tomcat.util.http.fileupload.impl.FileUploadBase.parseRequest(FileUploadBase.java:276)
at org.apache.catalina.connector.Request.parseParts(Request.java:2932)
... 31 more
08-Nov-2022 19:58:02.480 INFO [http-nio-8080-exec-4] org.apache.catalina.core.ApplicationContext.log HTTP/Manager: list: Listing contexts for virtual host 'localhost'
```

Figura D.14: Logs proporcionados por Tomcat

Para solucionar este problema se accede al archivo *apache-tomcat-9.0.68-webapps-manager-WEB-INF* y se modifican las siguientes líneas D.15 aumentando el número que se indica.



```

<request-character-encoding>UTF-8</request-character-encoding>

<servlet>
  <servlet-name>Manager</servlet-name>
  <servlet-class>org.apache.catalina.manager.ManagerServlet</servlet-class>
  <init-param>
    <param-name>debug</param-name>
    <param-value>2</param-value>
  </init-param>
</servlet>
<servlet>
  <servlet-name>HTMLManager</servlet-name>
  <servlet-class>org.apache.catalina.manager.HTMLManagerServlet</servlet-class>
  <init-param>
    <param-name>debug</param-name>
    <param-value>2</param-value>
  </init-param>
  <!-- Uncomment this to show proxy sessions from the Backup manager or a
  StoreManager in the sessions list for an application
  <init-param>
    <param-name>showProxySessions</param-name>
    <param-value>true</param-value>
  </init-param>
  -->
  <multipart-config>
    <!-- 50MB max -->
    <max-file-size>524288000</max-file-size>
    <max-request-size>524288000</max-request-size>
    <file-size-threshold>0</file-size-threshold>
  </multipart-config>
</servlet>
<servlet>
  <servlet-name>Status</servlet-name>
  <servlet-class>org.apache.catalina.manager.StatusManagerServlet</servlet-class>
  <init-param>
    <param-name>debug</param-name>
    <param-value>0</param-value>
  </init-param>
</servlet>
<servlet>
  <servlet-name>JSPProxy</servlet-name>

```

Figura D.15: Logs proporcionados por Tomcat

Cuando quise ejecutarlo mediante Eclipse no me dejaba añadir el proyecto al servidor de *tomcat*, indicando que las versiones no eran compatibles. Por ello se ha entrado en las propiedades del proyecto y se ha cambiado la versión del parametro *Dynamic Web Module* a la 3.1 en el apartado *Project Facets* como se aprecia en [D.16](#).

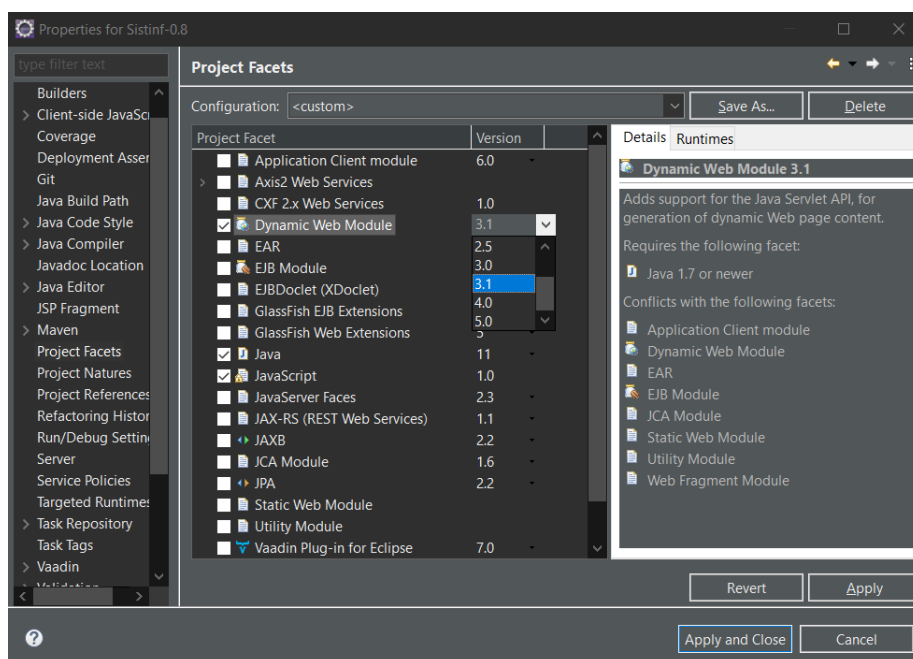


Figura D.16: Cambio de versión Dynamic Web Module

Alternativa a Heroku

A partir del día 28 de noviembre Heroku dejará de ser gratuito, pero ofrecen una alternativa para estudiantes. Esta opción es un acuerdo entre el [programa de estudiantes de GitHub](#) y Heroku. Esta colaboración viene explicada en la plataforma de [Heroku](#).

Por ello mismo migraremos nuestro proyecto a esa versión de Heroku ya que GitHub estudiantes nos proporciona una serie de créditos mensuales durante un año para desplegar nuestra aplicación.

Los pasos a seguir son los siguientes:

- Obtener la verificación GitHub para estudiantes en <https://education.github.com/students>, ver imagen [D.17](#).
- Seguir las instrucciones tras entrar en *Sign up for Global Campus* como se indica en [D.18](#).
- Indicar la escuela/universidad a la que pertenecemos y el uso que le vamos a dar a la cuenta, ver imagen [D.19](#).

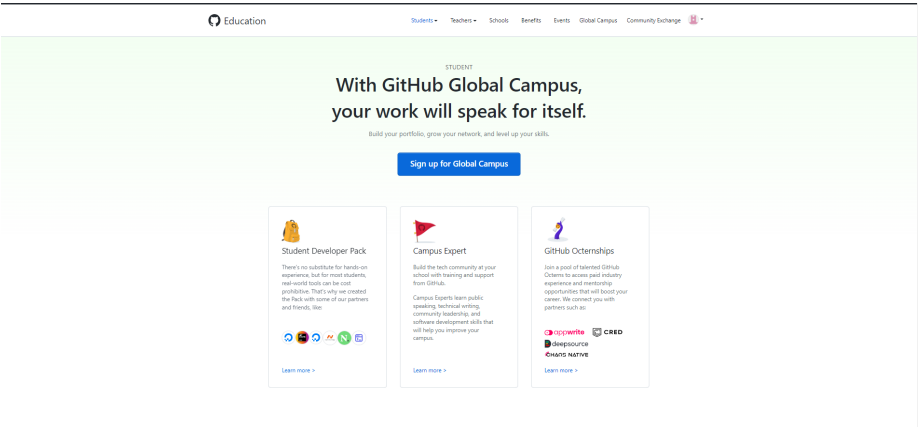


Figura D.17: Página de acceso a GitHub estudiantes

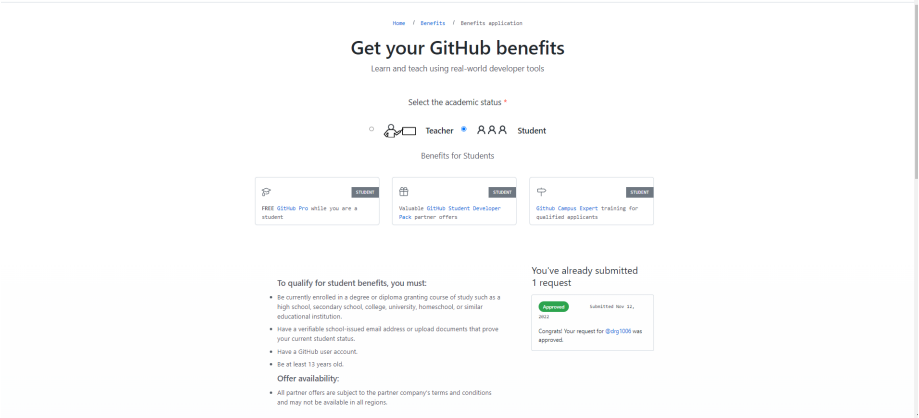
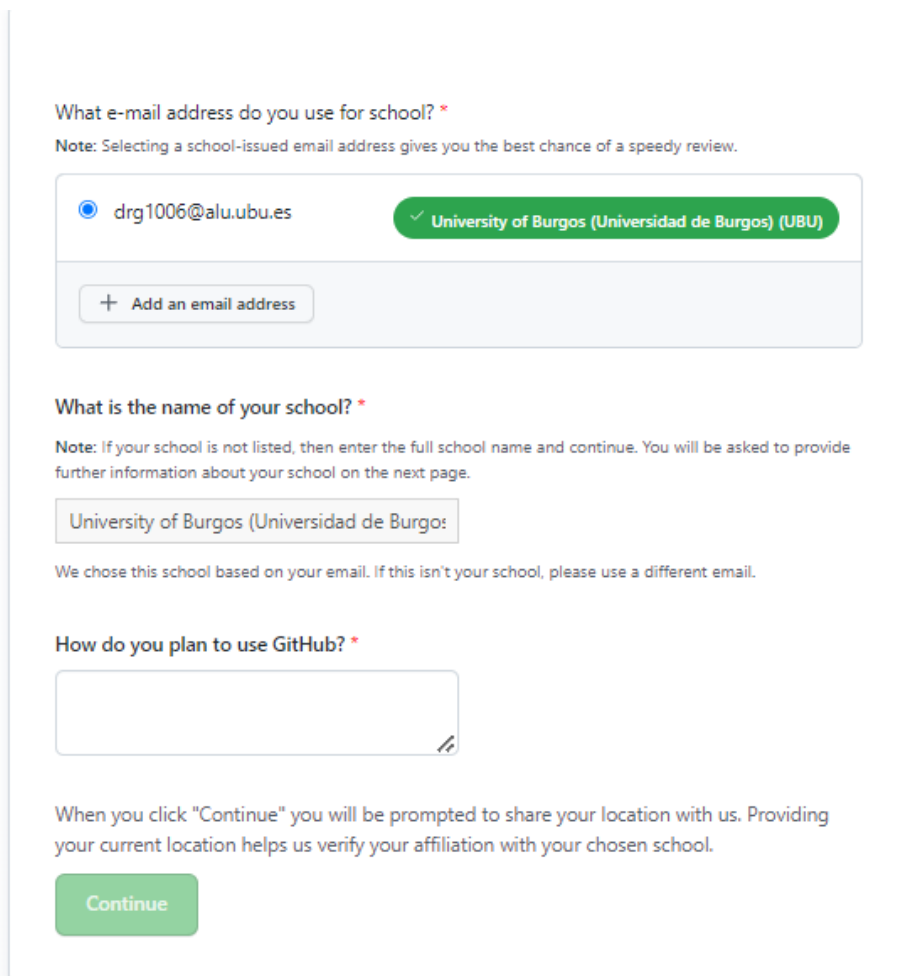


Figura D.18: Proceso de petición de GitHub for students



What e-mail address do you use for school? *

Note: Selecting a school-issued email address gives you the best chance of a speedy review.

☒ drg1006@alu.ubu.es ✓ University of Burgos (Universidad de Burgos) (UBU)

[+ Add an email address](#)

What is the name of your school? *

Note: If your school is not listed, then enter the full school name and continue. You will be asked to provide further information about your school on the next page.

University of Burgos (Universidad de Burgos)

We chose this school based on your email. If this isn't your school, please use a different email.

How do you plan to use GitHub? *

When you click "Continue" you will be prompted to share your location with us. Providing your current location helps us verify your affiliation with your chosen school.

[Continue](#)

Figura D.19: Aportar la información necesaria para la verificación

Tras ser verificados por GitHub debemos conectar nuestra cuenta con Heroku desde <https://www.heroku.com/github-students>.

Los pasos a seguir son los siguientes:

- Conectar nuestra cuenta de Heroku con la cuenta de GitHub para estudiantes.
- Debemos añadir una tarjeta bancaria, ya que vamos a utilizar un servicio de pago pero de forma gratuita, por lo que se solicitan esos datos (que se podrán retirar en un futuro sin ningún tipo de pago realizado).

- Esperar a que se confirme la solicitud realizada.
- Podemos comprobar si hemos sido verificados si tenemos los créditos añadidos en nuestra pestaña de *Account Settings* en *Billing* D.21.

El resultado final debería ser el siguiente D.20

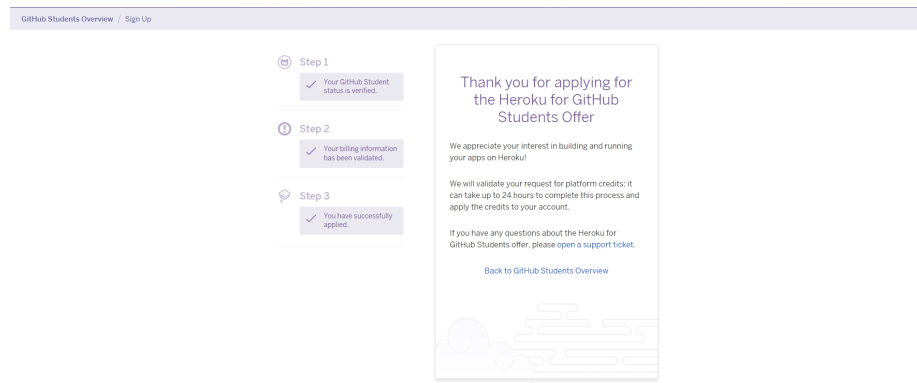


Figura D.20: Pestaña final tras aplicar a la oferta

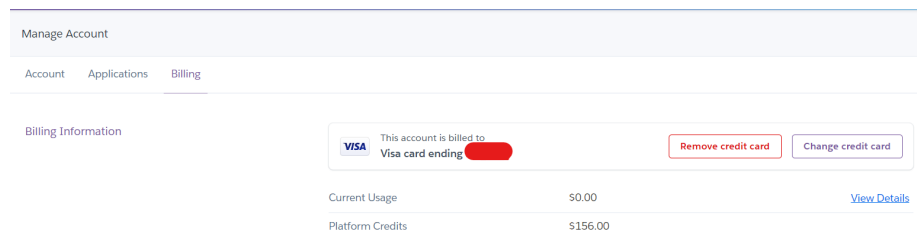


Figura D.21: Créditos de nuestra cuenta

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

E.1. Introducción

A continuación se describirán los requisitos mínimos a cumplir para que el usuario pueda entrar en la aplicación y usarla.

E.2. Requisitos de usuarios

Al estar la aplicación desplegada en <https://gestor-tfg-2022.herokuapp.com/> por lo que solamente hará falta disponer de Internet.

E.3. Instalación

Para utilizar la aplicación no será necesario instalar ningún componente en nuestro ordenador, a excepción de un navegador web.

E.4. Manual del usuario

A continuación se detallará el uso de la web, exclusivamente de las nuevas pantallas implantadas.

Histórico profesorado

En esta nueva pantalla implantada se expone información histórica sobre el profesorado.

En la parte superior de la pantalla tenemos una opción para actualizar la base de datos utilizada. Se informa de la última modificación de la base de datos actual y se le advierte al usuario que esta actualización es un proceso lento que puede tardar al rededor de un minuto. Ver imagen E.1.

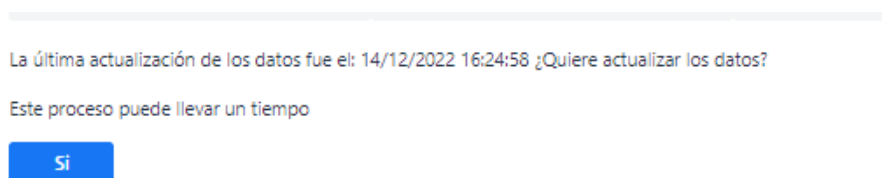


Figura E.1: Preguntar al usuario si quiere actualizar la base de datos

Si se ha optado por actualizar, se mostrará al final un aviso al usuario con el tiempo transcurrido durante la operación.

Posteriormente tenemos un pequeño apartado de información sobre el número de áreas, departamentos y profesores de la EPS. Ver imagen ??.

Información estadística

- Número total de profesores: 267
- Número total de areas: 23
- Número total de departamentos: 10

Figura E.2: Información sobre la EPS

A continuación debemos escoger las áreas, departamentos y profesores que deseamos visualizar en la gráfica. Una vez seleccionados *clickaremos* en actualizar gráfica. Si queremos seleccionar varios profesores debemos introducir primero uno y posteriormente el siguiente, para eliminarlos de la selección pulsaremos en la x del nuevo botón que se añade tras indicar un tutor. Ver imagen E.3.

GRÁFICA

☒ Seleccionar todos las Áreas

Áreas:

☐ Organización de Empresas ☐ Física Aplicada ☐ Matemática Aplicada ☐ Ingeniería Mecánica ☒ Lenguajes y Sistemas Informáticos ☒ Ciencia de la Computación e Inteligencia Artificial ☐ Expresión Gráfica en la Ingeniería ☐ Mecánica de Medios Continuos y Teoría de Estructuras
☐ Construcciones Arquitectónicas ☐ Ingeniería Eléctrica ☐ Química Orgánica ☐ Máquinas y Motores Térmicos ☐ Ingeniería e Infraestructura de los Transportes ☐ Expresión Gráfica Arquitectónica ☐ Edafología y Química Agrícola ☐ Ingeniería del Terreno
☐ Ingeniería de Sistemas y Automática ☐ Ingeniería Hidráulica ☐ Tecnología Electrónica ☐ Ingeniería de la Construcción ☐ Filología Inglesa ☐ Ciencia de los Materiales e Ingeniería Metalúrgica ☐ Urbanística y Ordenación del Territorio

☒ Seleccionar todos los departamentos

Departamentos:

☐ Ingeniería de Organización ☐ Física ☐ Matemáticas y Computación ☐ Ingeniería Electromecánica ☒ Ingeniería Informática ☒ Expresión Gráfica ☐ Ingeniería Civil ☐ Construcciones Arquitectónicas e Ingenierías de la Construcción y del Terreno ☐ Química ☐ Filología

Indique el profesor:

Álvar Arnaiz González

Figura E.3: Selección de parámetros

La gráfica muestra el número de TFGs por curso asignado a ese parámetro.

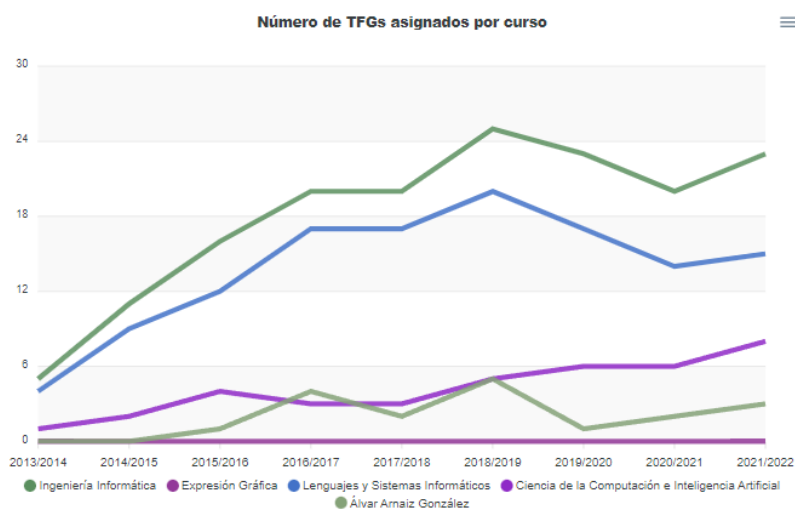


Figura E.4: Gráfica final tras seleccionar los parámetros

Creación de informes

En esta pantalla se le da al usuario la opción de añadir un área sobre el que se quiere hacer un informe, y el nombre que se le quiere dar. Ver imagen ??.

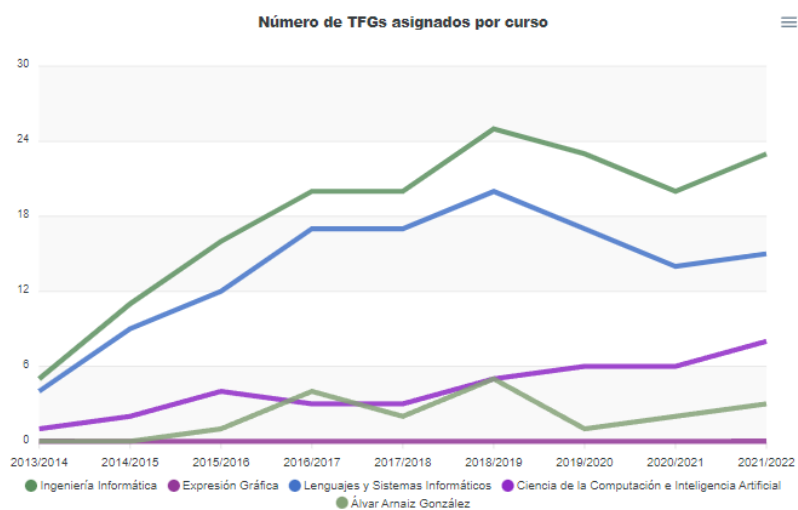


Figura E.5: Gráfica final tras seleccionar los parámetros

Este informe contendrá el número total de TFGs dirigidos, codirigidos y el número de créditos asignados a cada uno de los profesores del área seleccionado en el último curso académico. Si se seleccionan varias áreas se crearán varias hojas en el documento *excel* generado, con la información pertinente.

Oferta de TFG

Aceptación TFG

Bibliografía
