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 Diana Bringas Ochoa
en Universidad de Burgos — 1 de junio de 2021
Tutor: Álvar Arnaiz González y Carlos López Nozal

Índice general

Índice de figuras

Índice de tablas

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 de la legal.

A.2. Planificación temporal

Sprint 0 (26/10/2020 - 25/11/2020)

Puesta a punto del proyecto, planteamiento de las herramientas con las que trabajar, búsqueda de alternativas y toma de contacto con las herramientas nuevas que se van a emplear. Las tareas que se realizaron fueron:

- Añadir la extensión ZenHub al navegador Se añadió la extensión Zenhub al navegador de Google Chrome para utilizarlo en el GitHub.
- Clonar el repositorio en local. Mediante la aplicación GitHub Desktop se clono el repositorio del Gestor de TFGs mediante el enlace HTTP que proporciona GitHub.
- Investigar sobre Vaadin. A través de la página oficial de Vaadin se realizo la instalación y me informe acerca de Vaadin.
- Actualización del README.md Se modifico el README.md del proyecto para que reflejará los cambios respecto a la versión de la que se hizo fork.

- Investigar LaTeX Se procedió a buscar información sobre cómo instalar y manejar Latex para realizar la documentación del proyecto posteriormente.

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

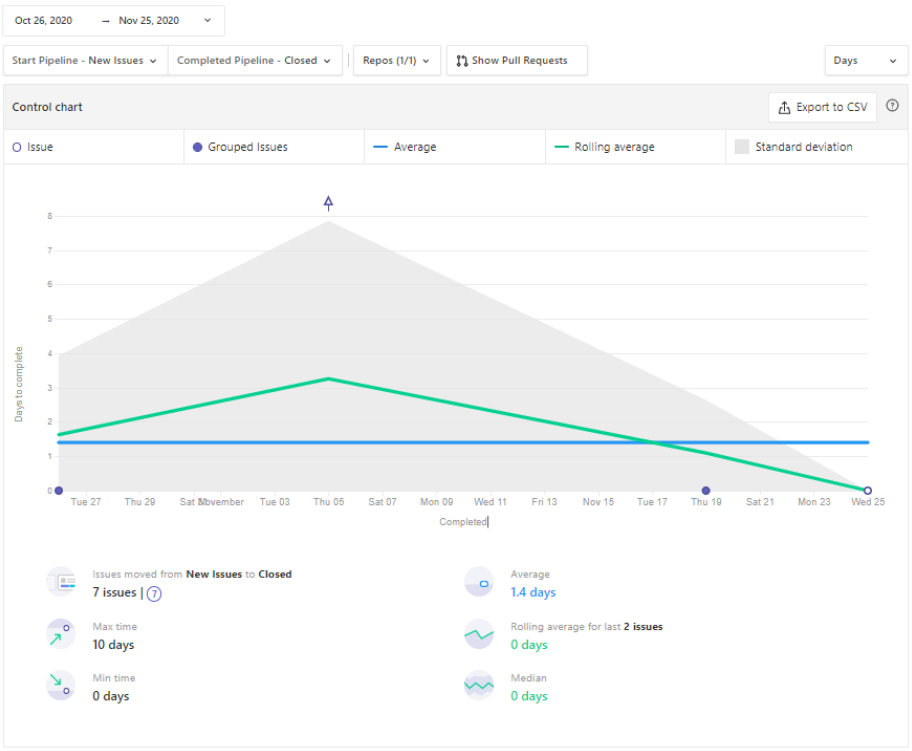

Figura A.1: Gráfica - Spring 0

Sprint 1 (25/11/2020 - 12/01/2021)

Generación de test unitarios, búsqueda de trabajos similares, cambio del driver para conectarse con el excel, información para obtener ideas de como realizar ciertas mejoras y comienzo de la documentación del proyecto. Mejora de la cobertura de la aplicación web.

Las tareas planteadas fueron:

- Instalación Miktex + TexStudio Tras la reunión del 25/11/20 se decicio que la mejor opción sería emplear un editor local, en vez del editor

online Overleaf. Para ello se instalo Miktex junto al editor de texto TexStudio.
- Se comienza la documentación en LaTeX - Spring 0 Creación de la documentación en LaTeX a partir de las plantillas.
- Generar nuevos test Para aumentar la covertura de la aplicación se definieron nuevos test.
- Cambiar driver JDBC Para poder usar los .ods se buscó una alternativa al driver para .csv que se empleaba.

Se puede ver el trascurso a través de la siguiente imagen A.2.

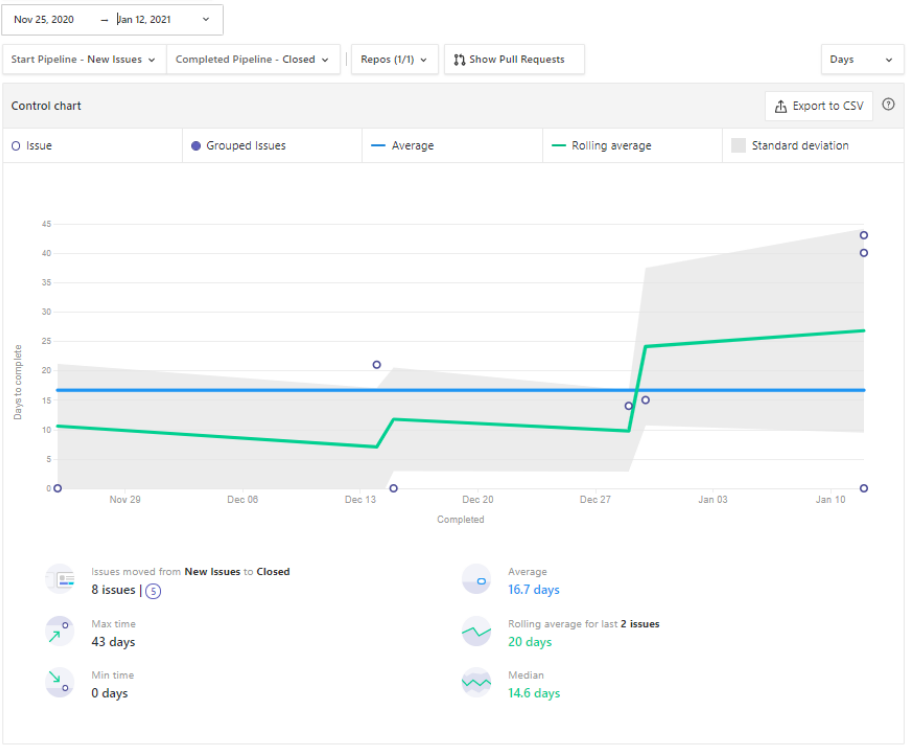

Figura A.2: Gráfica - Spring 1

Sprint 2 (12/01/2021 - 26/01/2021)

Búsqueda de nuevos driver que implementen JDBC para ficheros .xls. Prueba de opciones encontradas como Apache POI, SQLSheet, Fillo, Cdata JDBC for Excel y JdbcOdbcDriver. Integrar la API Fillo en el proyecto.

Las tareas planteadas fueron:

- Instalación del LibreOffice Se instaló Apache LibreOffice para manipular los fichero .xls y .csv que se emplean para la parte de datos de la aplicación.
- Cambiar de formato al fichero "**BaseDeDatosTFG.ods**". Se modificó el formato del fichero "**BaseDeDatosTFG.ods**" a "**BaseDeDatosTFG.xls**".
- Anexos - Modificación para poder cambiar el tamaño de las imágenes. Cambios en la plantilla de **anexos.tex** para poder modificar el tamaño de las imágenes al deseado.
- Probar el Apache POI Para verificar el funcionamiento de Apache POI, se incluyo en el proyecto de prueba **HolaMundoVaadin** y se realizaron varios ejemplos de prueba tomando como referente el proyecto principal.
- Error al intentar ejecutar la imagen del proyecto gabrielstan/gestion-tfg Bug realizado al intentar desplegar el proyecto gabrielstan/gestion-tfg.
- Desplegar proyectos relacionados con la gestión de TFG Proceso por el cual se probó a desplegar los proyectos relacionados con **GII 20.09 Herramienta web repositorios de TFGII**.
- Memoria - Mejoras Se realizaron varios cambios en la memoria.
- Incorporación del driver para fichero Excel Prueba de la API Fillo en el proyecto de prueba **HolaMundoVaadin**, y tras verificar su funcionamiento se procedió a incluirlo en el proyecto principal.
- Modificación de los nombres de los fichero csv Se cambiaron los nombres de los ficheros .csv para que coincidieran con los nombres de las hojas del fichero .xls.
- Memoria - Documentación Spring 2 Comienzo de la documentación de lo realizado en el Spring 2. En el cual se detallan las tareas realizadas.
- Modificaciones en los test Para probar la incorporación de la API Fillo se modificaron el test "**SintInfDataTest**".
- Realización de cambios para el funcionamiento de la nueva conexión para los fichero .xls Para poder usar la API Fillo se han tenido que realizar multitud de cambios en el proyecto. Aunque tanto el "**CsvDriver**" y "**Fillo**" emplean lenguaje SQL, "**CsvDriver**" emplea JDBC puro mientras que "**Fillo**" usa funciones y clases propias.

La siguiente imagen A.3 muestra cómo se han desarrollado las tareas a lo largo del tiempo.

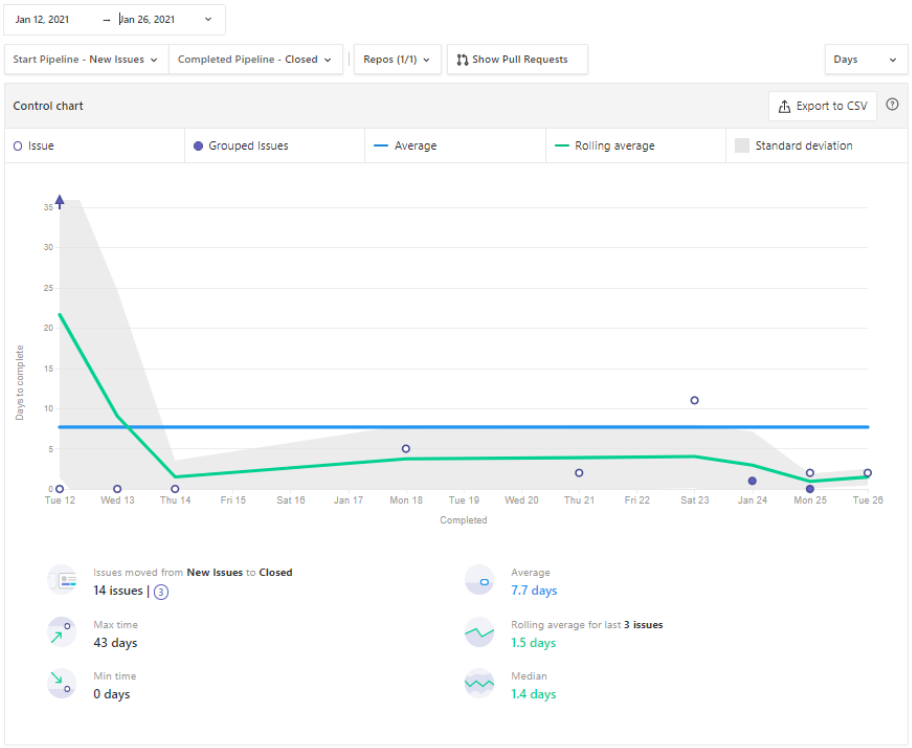

Figura A.3: Gráfica - Spring 2

Sprint 3 (26/01/2021 - 11/02/2021)

Las tareas planteadas fueron:

- Investigar opciones de hosting para el despliegue Se comparan varias opciones gratuitas para desplegar la aplicación, las cuales son: GitHub Pages, LucusHost, Awardspace, RunHosting, FreeHostia, X10Hosting y Heroku. Eligiendo la opción de Heroku, entre otras razones, porque proporciona la opción de conectar el despliegue con GitHub.
- Rediseño fachada y vistas Se elimina la parte vinculada a los datos de las vistas incluyéndola en a las clases fachada.
- Cambiar nombres a inglés. Se traducen los nombres de las variables, métodos y clases a inglés.
- Actualización de la Memoria y Anexos. Se realizan cambios y ampliaciones en el contenido de la documentación, en la Memoria y el Anexo.
- Instalación Heroku CLI Para realizar el despliegue del proyecto se instala el terminal de Heroku, denominado Heroku CLI. La instalación

se llevo a cabo según el tutorial de instalación de la página oficial de Heroku (https://devcenter.heroku.com/articles/heroku-cli).

- Incorporación del patron Factory Se crea una nueva clase con la función de seleccionar el tipo de acceso de datos, ya sea la clase fachada encargada de los ficheros .csv o, la que gestiona los ficheros .xls.
- Creación branch de prueba Rama del repositorio donde se almacena el proyecto de prueba formularioVaadin, el cual posteriormente será usado para realizar una prueba de despliegue en Heroku.
- Modificación de Test JUnit Se cambian los test para que sirvan para las funciones de la clase fachada para los ficheros .xls.
- Probar el despliegue del proyecto de prueba Se emplea el proyecto de prueba, formularioVaddin, para realizar una prueba de despliegue en Heroku. Se intentará realizarlo tanto, a través de Heroku CLI como, mediante la interfaz la página de Heroku.

En la siguiente imagen se enseña gráficamente el desarrollo de las issues A.4.

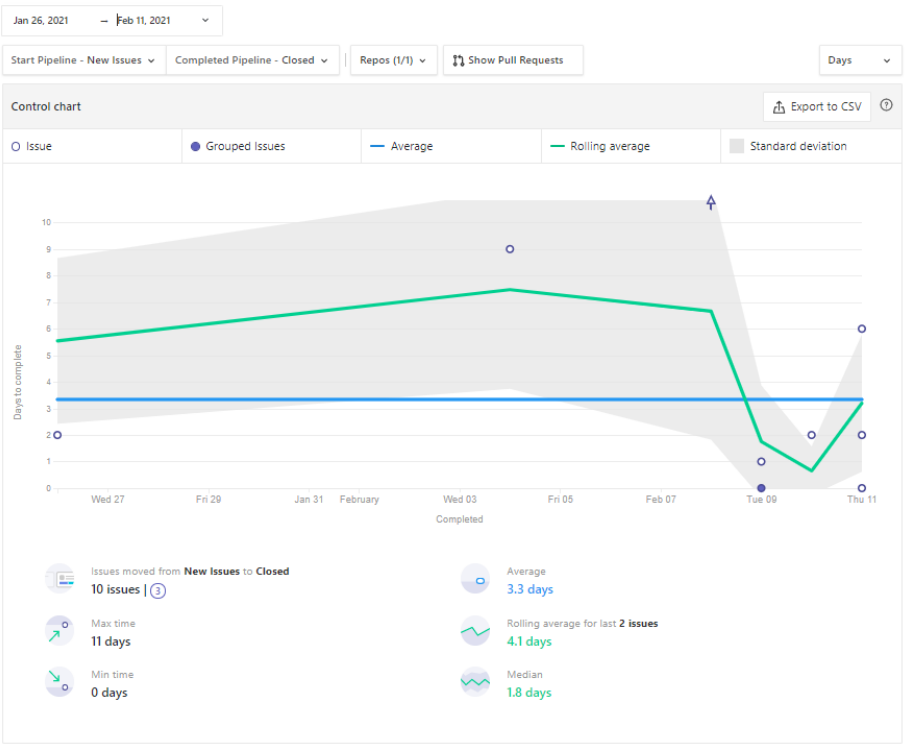

Figura A.4: Gráfica - Spring 3

Sprint 4 (11/02/2021 - 23/02/2021)

Se intentará incorporar la integración continua. Realizar el despliegue de la aplicación con Heroku. Añadir una columna de rankings en el Histórico. Cambiar las notas del histórico a privadas.

Las tareas planteadas fueron:

- Realizar la Integración Continua. A través de la opción GitHub Actions se incorporará la opción de compilar y ejecutar los test cuando se realiza un cambio (ya sea un push o un pull request) en el repositorio. Tiene como objetivo detectar fallos eficazmente mediante la integración automática frecuentemente de un proyecto.
- Actualización bibliografía Se incorporan los enlaces de las páginas o datos bibliográficos que se han empleado hasta ahora en la realización del proyecto.
- Cambiar las notas del Histórico a privadas Se elimina las notas de la columnas de "Notas.[en] la tabla de Históricos.
- Añadir dependencia Apache Commons Math
- Crear ranking de notas en Histórico
- Actualización del despliegue del proyecto de prueba en Heroku
- Mejora de test JUnit

Sprint 5 (23/02/2021 - 09/03/2021)

Añadir de nuevas columnas en la tabla de Históricos para reemplazar la columna Notas, la cual será eliminada. Crear test para comprobar los datos de los ficheros XLS. Investigación y prueba de análisis de calidad de código con SonarCloud.

Las tareas realizadas fueron:

- Eliminación de la columna Nota del Histórico
- Crear columna del ranking total en el Histórico. Se creó una nueva columna en la tabla de la vista del Histórico para representar las notas en comparación al resto de ellas
- - Investigar sobre SonarCloud. Para realizar las métricas que posteriormente irán en el apartado de Métricas, se analizará la calidad del código de algunos de los proyectos realizados anteriormente. Para ello, se investigará como usar el software SonarCloud para realizar las mediciones de calidad. Consta de dos versiones, una de pago y otra gratuita, siendo está última la que se empleará. Al ser una versión gratuita los proyectos serán públicos.

- Crear columna del ranking por cursos en el histórico. Al igual que con el caso del ranking total, se añadirá una nueva columna en la tabla de Históricos que contenga el ranking de una nota con respecto al resto de notas de ese mismo curso escolar (1 de septiembre a 30 de junio).
- Probar a desplegar el proyecto en Linux. Al estar el proyecto desplegado en Heroku se podrá acceder a él a través de la url desde cualquier SO.
- Parsear ficheros CSV. Se añadirán nuevos test que verifiquen que no existan errores de formato, por ejemplo que el formato de las fechas sea el indicado
- Separación de la clase SintInfDataTest en SintInfDataTestXLS y SintInfDataTestCSV. Al añadir una nueva clase fachada para el tipo de datos XLS se necesita otra clase para testarla, por lo que se dividirá la clase SintInfDataTest en dos clases, SintInfDataTestXLS, encargada de verificar los archivos y funciones correspondientes a los datos en XLS, y SintInfDataTestCSV, la cual testará los ficheros CSV.
- Añadir botón quality gate SonarCloud. Se incorporará en el README.md del proyecto un acceso directo a la página donde figurarán los análisis de los proyectos en SonarCloud.
- Añadir botón despliegue Heroku. Al igual que con SonarCloud, se incluirá un acceso a la página donde se encuentra desplegado el proyecto en Heroku. Para realizarlo se siguió los pasos de la documentación de Heroku.
- Analizar el proyecto sistinf con SonarCloud. Se realizará el análisis automático del proyecto principal, en el cual no se incluye el análisis del código en Java ya que esté lenguaje no está soportado en SonarCloud.
- Incluir el análisis automático del proyecto en Sonarcloud. Se seguirá el siguiente tutorial de la página de Sonarcloud para que se realice el análisis cada vez que se realice un push.
- Anexos actualización – B-Requisitos. Se añadirá el apartado de requisitos a la documentación de LaTex
- Anexos actualización - A-Plan-Proyecto. Modificación de los Anexos con el apartado A-Plan-Proyecto.

Sprint 6 (09/03/2021 - 23/03/2021)

Análisis del proyecto poolobject del código en Java en SonarCloud. Se investigó y probó a realizar el login a través de distintos métodos.

Las tareas realizadas fueron:

- Análisis del proyecto de prueba poolobject con SonarCloud. Se trata de un proyecto en Java al igual que el del Gestor-TFG-2021. Para ello se realizará un fork del proyecto en github y se procederá a realizar el análisis especificando la ubicación del código fuente en Java y los archivos binarios ya que el análisis automático de SonarCloud no incluye Java.
- Pruebas de Login a través de Heroku. Se probó a acceder a la parte de actualización de ficheros a través del login y subir un fichero, pero, no se modifico los datos de las vistas ni la fecha de actualización.
- Prueba Login con la app desplegada con Tomcat. Al ejecutar la app manualmente desde el IDE Eclipse empleando como herramienta para desplegar el proyecto Tomcat, se modificó la fecha de actualización de subida de los ficheros pero no se actualización los datos de las vistas.
- Solucionar problema en la actualización de los ficheros csv. Examinar y arreglar la razón por la que los ficheros no se estaban actualizando y por tanto, no cambiaban los datos de las vistas. Para ello, se llevo a cabo varias modificaciones en las clases fachada de los datos.
- Cambiar configuración SonarCloud. Se excluirá los ficheros que no se desean examinar en el análisis de la calidad del código como los ficheros propios de Vaadin, los ficheros CSS, entre otros. Anteriormente el análisis que se había realizado sobre SonarCloud no estaba teniendo en cuenta los códigos en Java debido a que el análisis automático de SonarCLoud no es compatible con Java. Por lo que se deberá especificar donde se encuentra el código fuente que se desea examinar y los archivos binarios de Java.
- Investigar cómo realizar el Login a través de UBUVirtual. Uno de los requisitos es realizar un Login que permita autentificarse conel correo de la UBU o similares, por lo que se investigará que herramientas se pueden emplear para realizar esto. Finalmente la opción que se escogío fue Firebase, un servicio de backend que dispone de SDKs fáciles de usar y bibliotecas de IU ya elaboradas.
- Investigar cómo importar los ficheros CSV y XLS a Heroku. Se puede subir los ficheros mediante Amazon S3 (o otro almacenamiento en la nube que se pueda conectar con Heroku) y, a través de Skyvia importar los datos (volcarlos) en la base de datos (Heroku Postgresql). O simplemente subir los ficheros en el .war y cuando se actualicen modificarlos, esta será la opción que se empleará.
- Prueba - Login con Microsoft. Se probará el código de ejemplo para iniciar sesión mediante Microsoft en aplicaciones web en Java. Siguiendo el tutorial de la página de Microsoft. Para ello se registró la app en

Azure y se siguió el tutorial de ejemplo Azure. No se consiguió realizar el login ya que se necesitan permisos los cuales no dispongo.
- Añadir más extensiones a gitignore. Para evitar que se suban ficheros no deseado se incluyeron más extensiones a ignorar en el fichero gitignore.

Sprint 7 (23/03/2021 - 13/04/2021)

Análisis de la calidad del código de los proyectos presentados en 2020 con SonarCloud. Investigar cómo realizar el login con Firebase. Investigar y realizar la migración de versión de Vaadin.

Las tareas realizadas fueron:

- Forks de todos los proyectos presentados en 2020. Se les añadirá al principio del nombre ÜBU-TFG"para identificarlos.
- Instalar SonarCloud CLI. Se realizará la instalación según la documentación de SonarCloud
- SonarCloud - Analizar proyecto Gestión Aulas Informática
- SonarCloud - Analizar proyecto UBUMonitor Clustering
- SonarCloud - Analizar proyecto Medidor estadístico metajuego Magic The Gathering
- SonarCloud - Analizar proyecto TourPlanner-FrontEnd-Cliente
- SonarCloud - Analizar proyecto UBUVoiceAssistant
- SonarCloud - Analizar proyecto LogScope
- SonarCloud - Analizar proyecto PruebaNetExtractor
- SonarCloud - Analizar proyecto Reserva aulas informática
- SonarCloud - Analizar proyecto MetrominutoWeb
- SonarCloud - Analizar proyecto Sentinel
- SonarCloud - Analizar proyecto CENIEH and Ariadne
- SonarCloud - Analizar proyecto Plataforma de text mining sobre repositorios
- SonarCloud - Analizar proyecto UBUEstelas
- Investigar cómo migrar de versión de Vaadin
- SonarCloud - Analizar proyecto XRayDetector
- SonarCloud - Analizar proyecto Jellyfish Forecast
- SonarCloud - Analizar proyecto Análisis Comercial Urbano
- SonarCloud - Analizar proyecto Iris classifier
- SonarCloud - Analizar proyecto Asistente de programación C
- SonarCloud - Analizar proyecto Flutter Serpiente
- Instalación NodeJS

- SonarCloud - Analizar proyecto Blockchain en una cadena de distribución de productos
- Crear proyecto en Firebase
- Migración de versión de Vaadin
- SonarCloud - Analizar proyecto Estudio de herramientas para realidad aumentada
- SonarCloud - Analizar proyecto ARBUBU
- SonarCloud - Analizar proyecto Free Connect
- Memoria - Objetivos del proyecto
- Anexo - Especificación de Requisitos

Sprint 8 (13/04/2021 - 04/05/2021)

Se migrará el proyecto a Vaadin 14. Se creará la primera release (versión 0.6) con la app con la posibilidad de subir tanto .csv como .xls y su correspondiente actualización de las vistas.

Las tareas realizadas fueron:

- Revisión de memoria y anexos Realización de correcciones recomendadas por el tutor de la documentación en LaTeX.
- Actualización subida de ficheros Se llevaron a cabo varias modificaciones en el código para que se realizará correctamente la actualización de las vistas con los nuevos datos, tanto xls como csv.
- Incorporación Firebase para el Login
- Migración a Vaadin 14 Continuación del proceso de migración del proyecto a Vaadin 14 para lo cual se debió de investigar y sustituir múltiples elementos que ya no existían en la nueva versión.
- Creación Release 0.6 Primera versión con la aplicación en la versión con Vaadin 7 con la posibilidad de subir tanto ficheros csv como xls.

Sprint 9 (04/05/2021 - 11/05/2021)

Finalización del proceso de migración del proyecto a Vaadin 14 y despliegue de la aplicación en su nueva versión en Heroku.

Las tareas realizadas fueron:

- Investigar nueva API para realizar las gráficas del Histórico Al cambiar de versión de Vaadin, JFreeChart, el componente empleado para realizar las gráficas del histórico, deja de ser válido. Por lo que se realizar un búsqueda en Vaadin Directory de componentes que puedan

sustituirlo. Las dos mejores opciones encontradas son Vaadin Chart, la cual es de pago por lo que es descartada y ApexChart, la opción elegida.
- Continuar la migración a Vaadin 14 Se concluye la migración de la app Gestor-TFG-2021 a Vaadin 14.
- Actualización componentes de las Vistas Se implementaron diversas modificaciones para conseguir que fuese similar estéticamente a su versión anterior.
- Prueba del proyecto en Heroku
- Modificación Login

Sprint 10 (11/05/2021 - 18/05/2021)

Realizar despliegue en Java 11 con la nueva versión en Vaadin 14. Continuar con el nuevo login con Firebase. Introducir mejoras en el código.

Las tareas realizadas fueron:

- Pruebas de actualización de ficheros en el proyecto en el despliegue de heroku Se probará a actualizar varios ficheros csv y xls para comprobar que se realiza correctamente. También se realizo una prueba con el fichero XLS obtenido por los tutores en la anterior reunión para probar.
- Despliegue del proyecto en Vaadin 14 en Heroku
- Actualizar obtención ranking por cursos La obtención de los cursos para el ranking se obtenía de la vista del proyecto (N2-Proyecto). Se cambiará para que obtenga el curso a partir de la fecha de presentación y asignación del Histórico ($N3_Historico$).
- Realizar mejoras en el código. Se realizará diversas mejoras cómo la introducción de más información en el logger, actualización de los filtros empleados en las tablas (Grid), eliminación warnings y imports no usados.
- Cambio de versión a Java 11 en el workflow Maven CI/CD En los últimos commits no se pasaron los test de la Integración continua a que se cambio la aplicación a Java 11, en el pom.xml, mientras que en el workflow sigue estando la 8. Esto se solucionará cambiando la versión de Java (java-version) en el workflow, en el fichero github-ci.yml concretamente.
- Creación release 0.8 Creación nueva release con la aplicación en Java 11 y Vaadin 14.
- Actualizar styles del proyecto Modificaciones estéticas de la aplicación para conseguir un resultado más similar al de la versión anterior de la

aplicación, con Vaadin 7. En estos cambios destacan por ejemplo el cambio del estilo del texto, tablas o títulos.

Sprint 11 (18/05/2021 - 01/06/2021)

Las tareas realizadas fueron:

- Continuar con Login con Firebase
- Actualizar README Actualizar enlace al despliegue de Heroku y introducir más información acerca de la app y cómo ejecutarlo.
- Añadir reglas de seguridad en Firestore Tras crear la base de datos Firestore se añadirán nuevas reglas de seguridad para impedir su modificación a usuarios no permitidos.
- Añadir iconos faltantes en la app Se buscó iconos equivalentes a los que anteriormente había en las vistas de la aplicación, ya que al actualizar de versión de Vaadin ya no existen.
- Actualizar Footer Añadir iconos en el Footer, los nombres faltantes (y sus respectivos correos de la ubu) y revisar fecha actualización ficheros. Se agregó, además de la fecha de actualización de los ficheros CSV, la fecha de actualización del archivo XLS.
- Renombrar ficheros XLS al subirse Se modificó la lógica de la vista de actualización de ficheros (UploadView) para que se pudiese subir el fichero XLS con cualquier nombre y fuese la propia app la que lo renombrará con el nombre requerido. En el caso de los ficheros csv, se indicará en la vista de UploadView cómo deben llamarse: $N1_{D}ocumento$, $N1_{N}orma$, $N1_{T}ribunal$,$N2_{A}lumno$,$N2_{P}royecto$ y $N3_{H}istorico$.
- Añadir seguridad con Spring boot Para evitar que se pueda acceder a la vista de la actualización de ficheros sin haber iniciado sesión previamente se comenzó introduciendo Spring boot security pero, finalmente se encontró una forma más sencilla y se descarto el uso de esta opción.
- Modificar UploadView para verificar si hay un usuario logeado Se añadió un método que comprueba y controla, antes de entrar al evento de UploadView, si hay algún usuario logeado. En caso contrario, redirige al login para que el usuario inicie sesión.

A continuación, se expondrá una ilustración donde se aprecia el trascurso del Spring y el desarrollo de las tareasA.5.

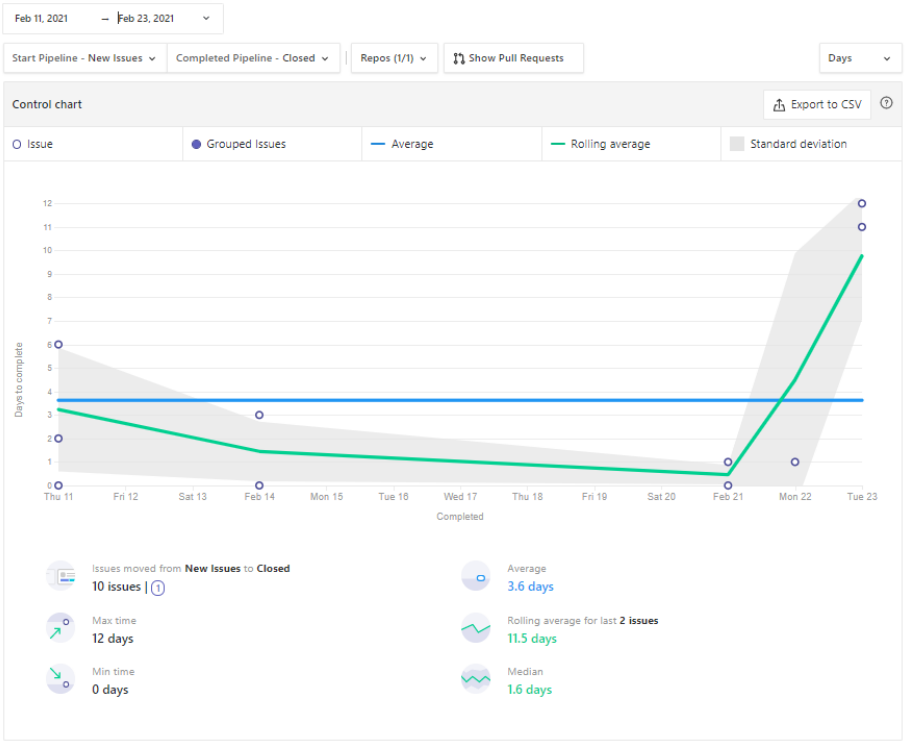

Figura A.5: Gráfica - Spring 4

A.3. Estudio de viabilidad

Viabilidad económica

En este apartado se detallan los costes que llevaría realizar este proyecto.

Coste del personal

Coste hardware

Referente a los costes del equipo utilizado en el desarrollo del trabajo. Teniendo en cuenta el precio del ordenador empleado de aproximadamente 700 euros.

Coste software

Referente a los costes de las herramientas software no gratuitas empleadas en el proyecto. Como es el caso del Sistema Operativo Windows o el Microsoft Office 365.

Viabilidad legal

En este apartado se detallaran las licencias de cada dependencia que se ha utilizado en el proyecto

Software	**Licencia**
Vaadin	Apache License 2.0
Vaadin Maven Plugin	Apache License 2.0

Tabla A.1: Dependencias del proyecto

Apéndice B

Especificación de Requisitos

B.1. Introducción

Los requisitos[**?**] del proyecto son los puntos clave que se deben cumplir.

B.2. Objetivos generales

El objetivo del proyecto es la mejora de la aplicación Web centrándose en los siguientes puntos:

- Incorporación y validación de un nuevo tipo de datos (XLS)
- Realización de la métricas con los resultados del análisis de la calidad del código de los TFG (SonarCloud)
- Mejoras en la estética de la aplicación Web.
- Modificación del Login para conectarse con la cuenta de la Universidad.

B.3. Catalogo de requisitos

Se describirán los requisitos específicos, funcionales y los no funcionales.

B.4. Especificación de requisitos

Requisitos funcionales

- **RF-1 Adicción nueva capa de datos**: la aplicación permite emplear dos tipos de archivos, xls y csv, como entrada de datos.
  - **RF-1.1 Subida de datos:** el usuario que tenga permisos podrá subir archivos de datos, tanto en el formato xls como en el csv.
- **RF-2 Login**: el sistema permitirá la subida de datos a ciertos usuarios que se autentificarán a través del login de la aplicación.

Requisitos no funcionales

- **RNF-1 Seguridad**: la aplicación solamente debe permitir la subida de datos a los usuarios con permisos.
- **RNF-2 Mantenibilidad**: mejora de la aplicación para permitir la escalabilidad y incorporación de nuevas modificaciones en el futuro de forma sencilla. Un ejemplo de esto sería la actualización de Vaadin a una versión nueva para conservar el soporte en el futuro.
- **RNF-2 Mejora diseño**: se realizarán mejoras para que la aplicación sea más atractiva e informativa. Se optará siempre por opciones intuitivas y sencillas de utilizar.

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

En esta sección se explica la estructura del proyecto, el proceso de instalación del framework y herramientas necesarias para desarrollar el trabajo, cómo realizar la compilación, instalación y ejecución del proyecto y las pruebas que se han llevado a cabo.

D.2. Estructura de directorios

D.3. Manual del programador

A continuación se explicará cómo realizar la instalación de los programas necesarios para el desarrollo de la aplicación.

Instalación de Java

Como en el proyecto se usa Vaadin 8, se debe emplear la **versión 8 de Java**, en concreto se ha usado la jdk1.8.0_271.

Para ello se deberá ir a la página de descargas de Oracle Java SE 8.0 y descargar la versión de JDK 8 correspondiente con tu sistema operativo y su arquitectura, ya sea de 64 o 32 bits. Ver imagen D.1.

Tras escoger la versión según nuestro SO se deberán de leer y aceptar las licencias de uso de Oracle, como se muestra en la figura D.2, y dar a descargar.

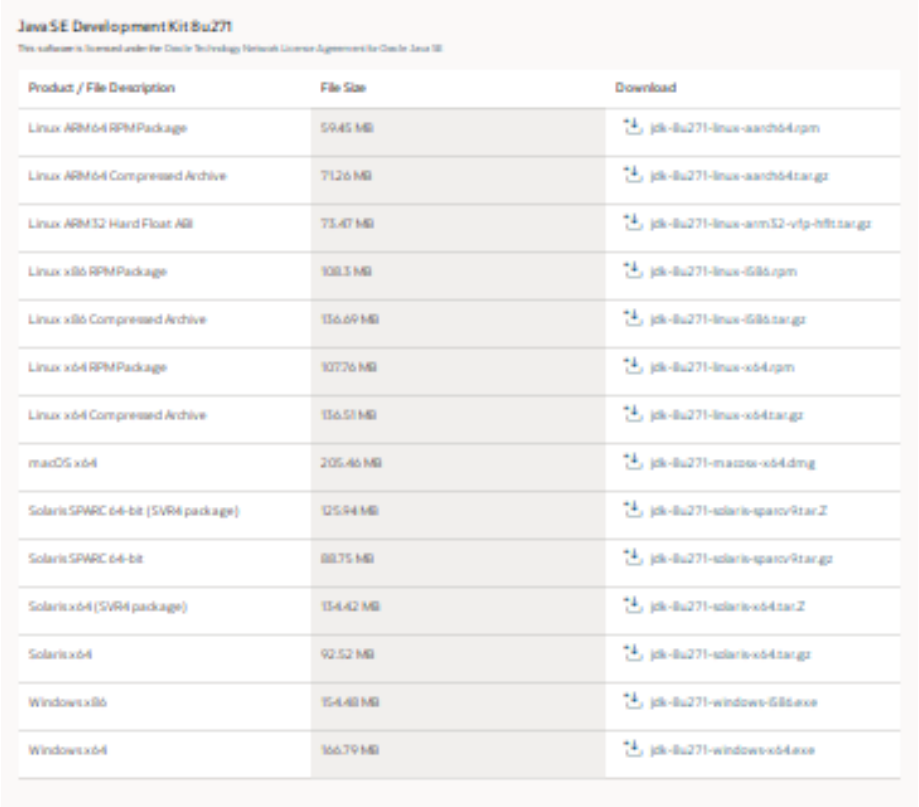

Figura D.1: Descarga de JDK 8

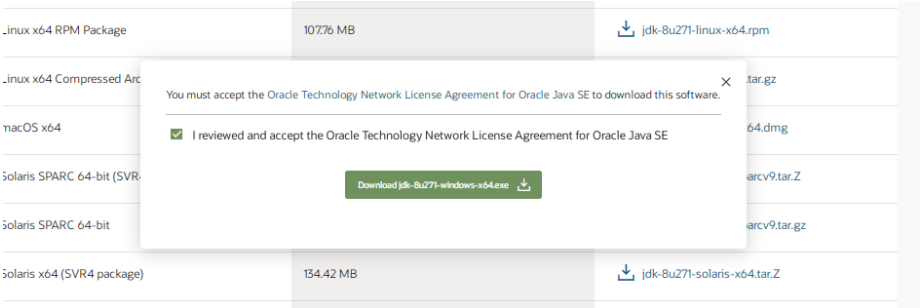

Figura D.2: Descarga JDK 8 Licencia

Una vez descargado, se deberá ejecutar el instalador y seguir el proceso de instalación del asistente.

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 2020-06.

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 2020-06 R**. Ver imagen D.3.

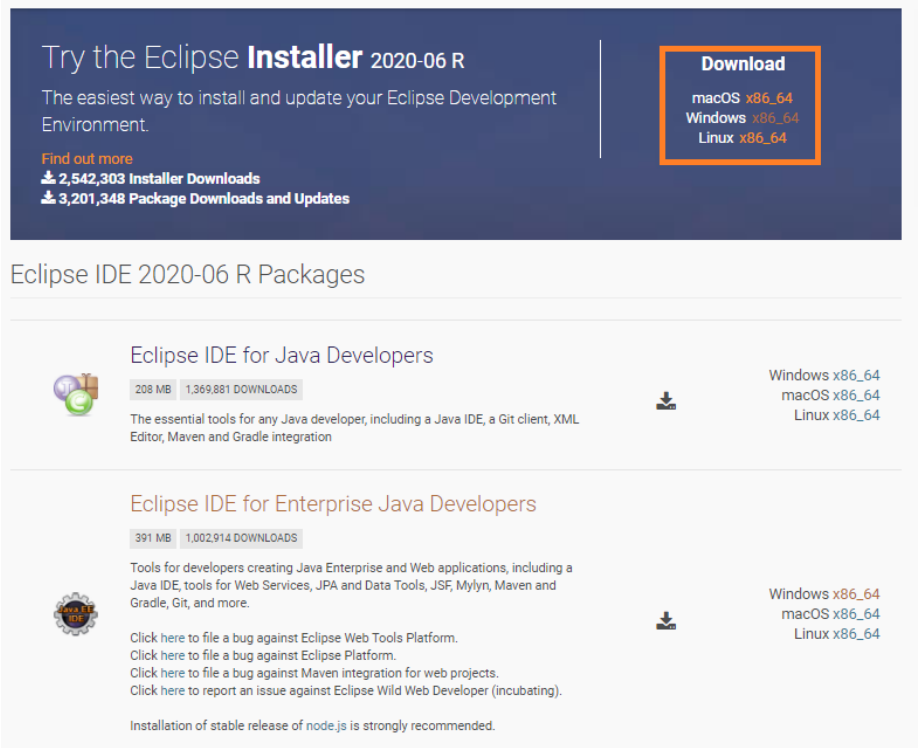

Figura D.3: Descargar IDE Eclipse

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***".

En el siguiente paso, en el apartado de "**Java 1.8 + VM**" se deberá seleccionar la carpeta donde se encuentra el JDK 8, instalado anteriormente.

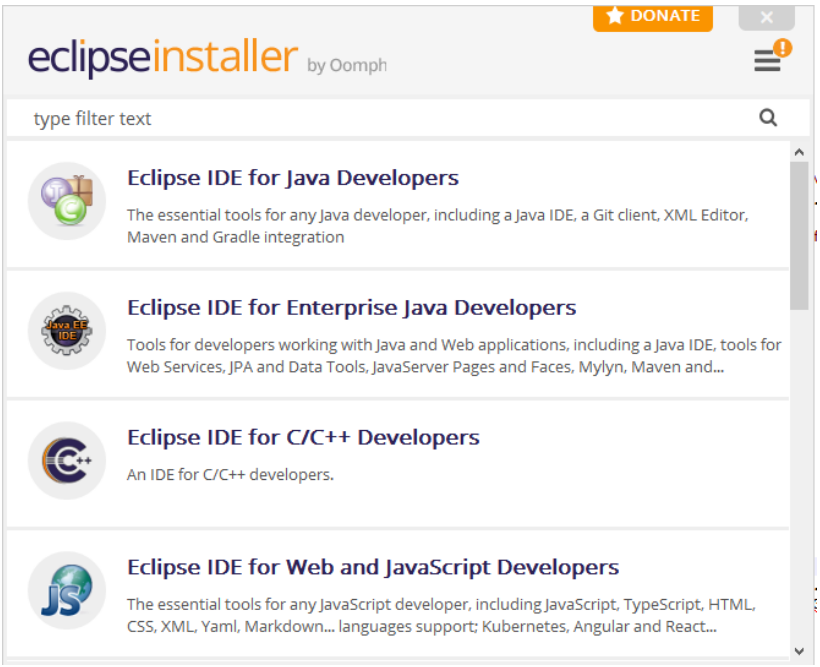

Figura D.4: Seleccionar Eclipse

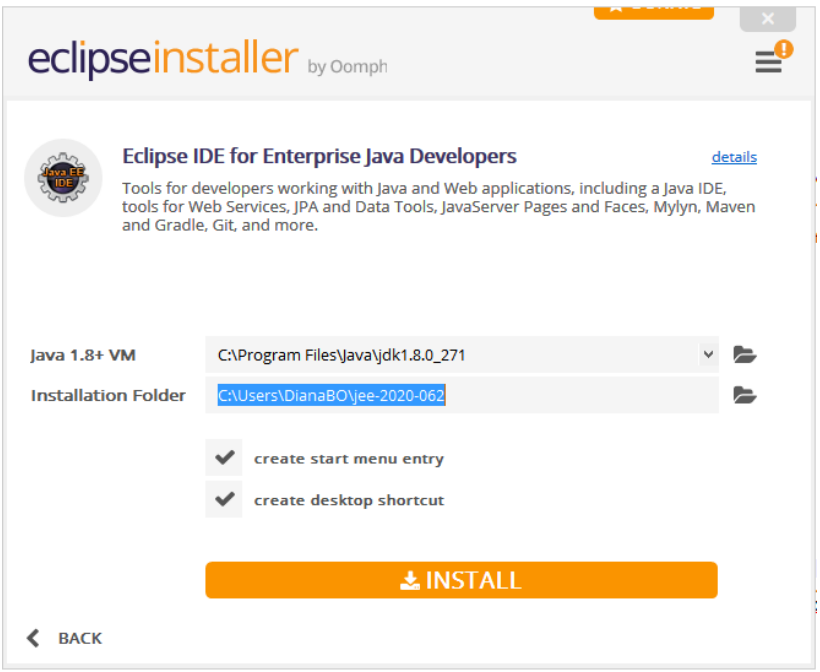

Figura D.5: Seleccionar JDK que usará el IDE

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**, el cual se encuentra en la opción de "***Help/Eclipse Marketplace...***" de la barra de herramientas.

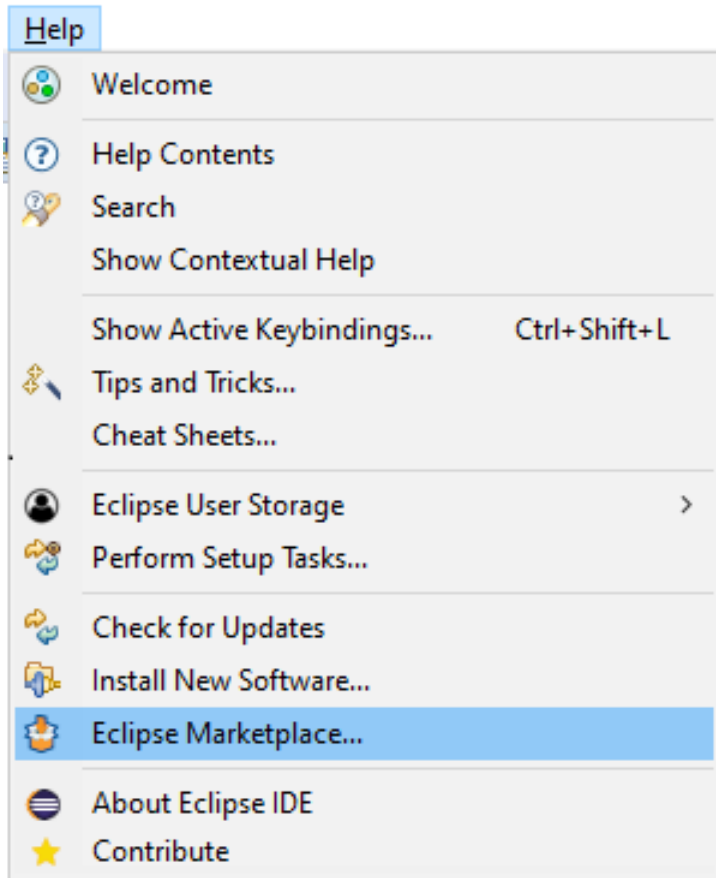

Figura D.6: Eclipse marketplace

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.

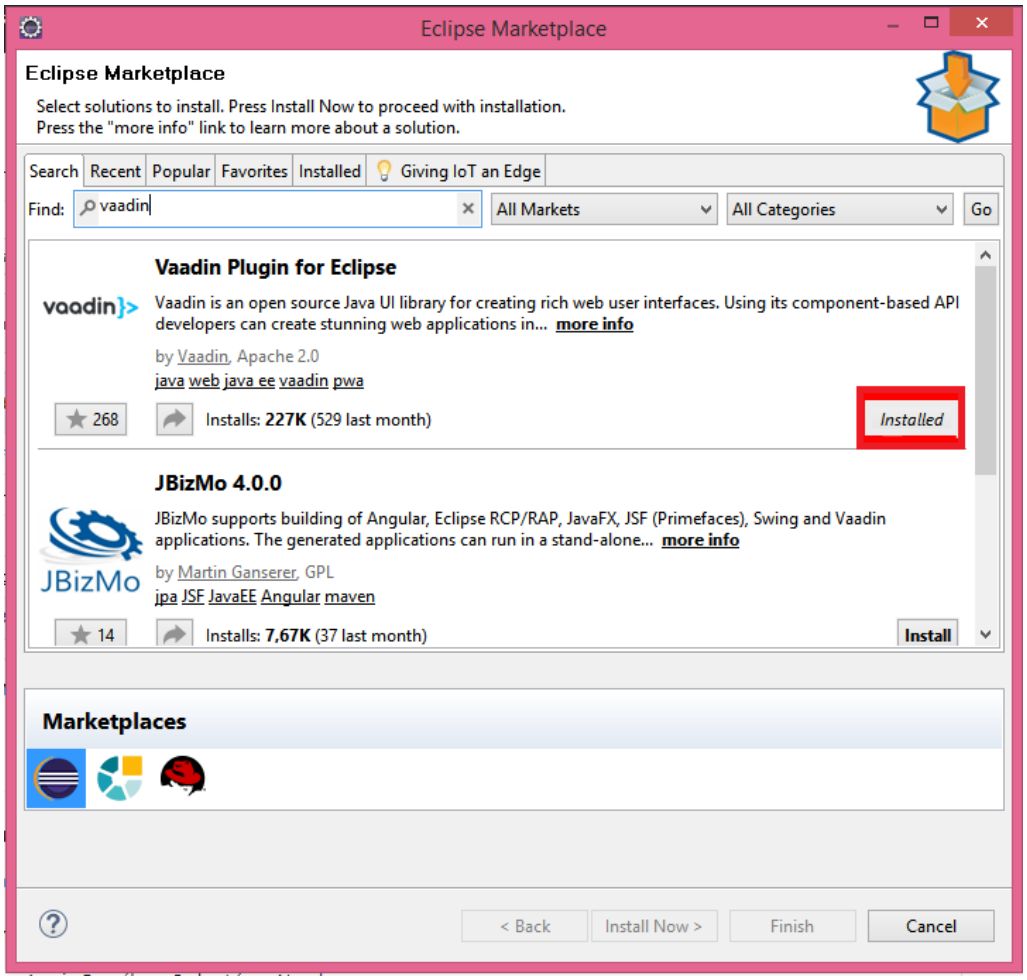

Figura D.7: Plugin Vaadin

D.4. Compilación, instalación y ejecución del proyecto

Se explicará como compilar, instalar e ejecutar el proyecto. En el caso de la ejecución, se detallará como hacerlo con el términal de Windows 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.

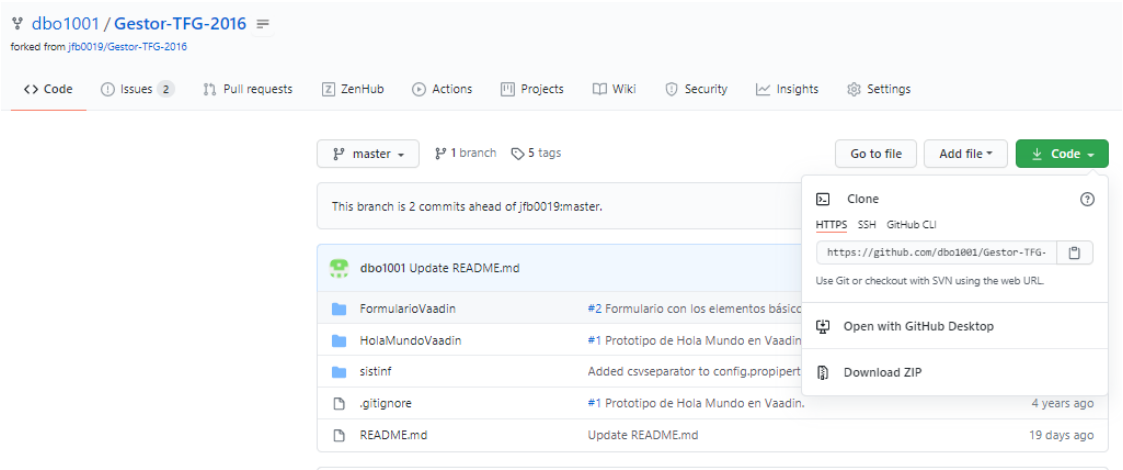

Figura D.8: Copiar URL repositorio

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

Para abrir el proyecto con Eclipse se seleccionara en la barra de herramientas ***File/Import...***. Aparecerá una ventana en la que se optará por la opción "***Projects from Folder or Archieve***" y se hará click en "***Next***".

Después se hará click en "***Directory...***" y se seleccionará la carpeta del proyecto con nombre "**sistinf**" y términaremos la importación con "***Finish***".

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

No se requiere ninguna instalación por parte del usuario, simplemente puede acceder a la aplicación web a través del enlace .

E.4. Manual del usuario

Bibliografía