Instituto Tencológico de Costa Rica

Avance 2

Esteban Quirós Alvarado - 2014012892
Mauricio Martínez López - 201012121
Jason Latouche Jiménez - 2015146294

Ingeniería en Computación
Aseguramiento de Calidad de Software
Profesor: Saúl Calderón Ramírez
Abril 2018

Índice

1. Introducción

Para este segundo avance, se especificarán otros aspectos conforme al diseño, prioridad de requerimientos, explicación de el uso de algunas herramientas para medir y validar ciertos aspectos del proyecto, la modificación de algunos aspectos del SyRS, la realización de otras nuevas pruebas unitarias y la implementación de tres requerimientos.

2. Necesidades prioritarias

A continuación se presenta una tabla de los requerimientos más prioritarios para esta etapa del proyecto:

Cuadro 1: Tabla de prioridades

Req #	**Descripción**	**Prioridad**
1.6.1.1	Permisos y acceso	Media
1.6.1.2	Almacenar datos del paciente	Media
1.6.1.3	Cargar datos del paciente	Media
1.6.1.4	Subir una imagen asociada a un paciente	Alta
1.6.1.5	Mostrar predicción	Alta
1.6.1.6	Subir nuevas imágenes	Alta
1.6.1.7	Almacenar predicciones	Media
1.6.2.1	Interfaz de usuario	Alta
1.6.2.2	Predecir edad en menos de 30s	Baja
1.6.2.3	Almacenar bitácora de cambios	Baja
1.6.2.4	Tiempos requeridos	Baja

Cuadro 2: Validación de requerimientos frente a elementos de diseño

Requerimientos #	**Diseño #**
1.6.1.1	3.1
1.6.1.4 - 1.6.1.6	3.2
1.6.1.5 - 1.6.1.7	3.3
1.6.2.1	3.4
1.6.1.2 - 1.6.1.3 - 1.6.2.1	4.5

Cuadro 3: Validación de elementos de diseño frente a implementaciones

Diseño #	**Implementación**
3.1	Encriptación de las contraseñas
3.2	Subir imagen
3.3	Predecir edad
3.4	Estructura web
3.5	Implementación de componentes

3. Validación

3.1. De requerimientos

3.2. De implementación

4. Unidades de diseño

4.1. Inicio de sesión

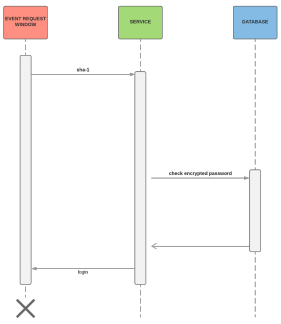

Figura 1: Diagrama de secuencia para el inicio de sesión

4.2. Componentes para subir una imagen

Figura 2: Diagrama de componentes para subir una imagen

4.3. Predecir edad

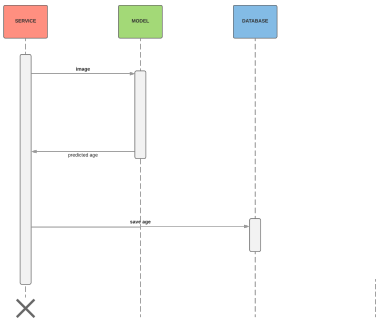

Figura 3: Diagrama de secuencia para calcular la edad

4.4. Pantalla de subir imagen

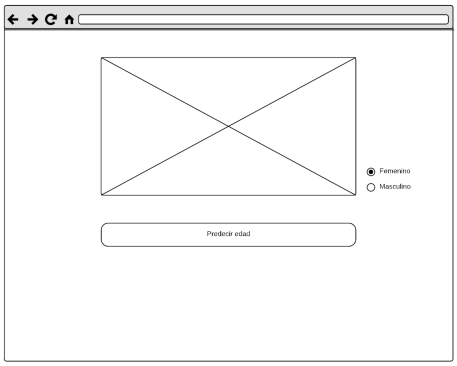

Figura 4: Wireframe para subir una imagen

4.5. Componentes del sistema

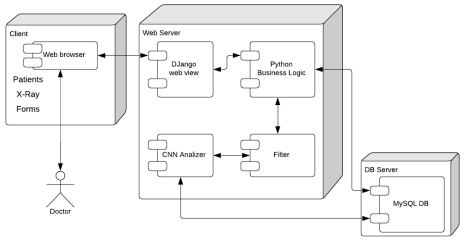

Figura 5: Diagrama de componentes para el sistema

5. Uso de Herramienta de Verificación de la Codificación

Para el proyecto, escogimos el estándar de codificación PEP-8, el cuál tiene una serie de características que creemos son las mejores para este proyecto. Para que este estándar de codificación se cumpla en su totalidad durante el proyecto, es necesaria una herramienta con la que se puede verificar que dicho estándar se está cumpliendo en todo el código, en nuestro caso al usar Eclipse ambientado a Python, este se puede configurar para que todo el código escrito siga el estándar PEP-8, por lo que a continuación explicaremos los pasos de como es que se activa y utiliza dicha funcionalidad en la herramienta Eclipse.

*Se asume que la configuración del ambiente PyDev en la herramienta Eclipse ha sido previamente realizada con éxito.

El primer paso en la ventana principal es dar click en la parte superior de la ventana, en la opción de Window ("Ventana") y luego en Preferences ("Preferencias"), la cuál nos abrirá otra ventana.

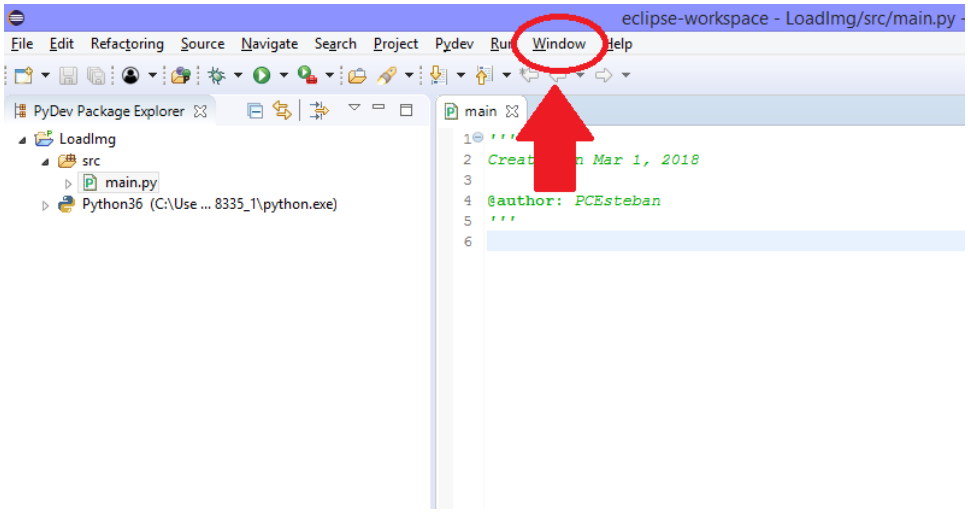

Figura 6: Primer Paso

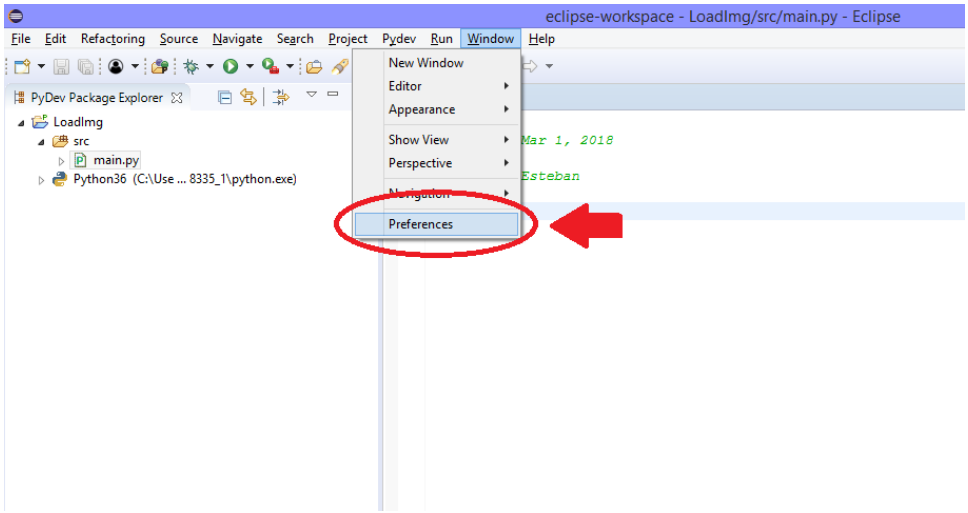

Figura 7: Segundo Paso

Luego, en la ventana de preferencias que se nos presenta, al lado izquierdo se nos desplegarán varias opciones, en la cual daremos click en la de **PyDev**, la cuál abrirá más opciones, entre las que está la opción de **Editor**, en la que también daremos click para que luego nos dé más opciones, de la cuales escogeremos **Code Analysis** (Análisis de Código).

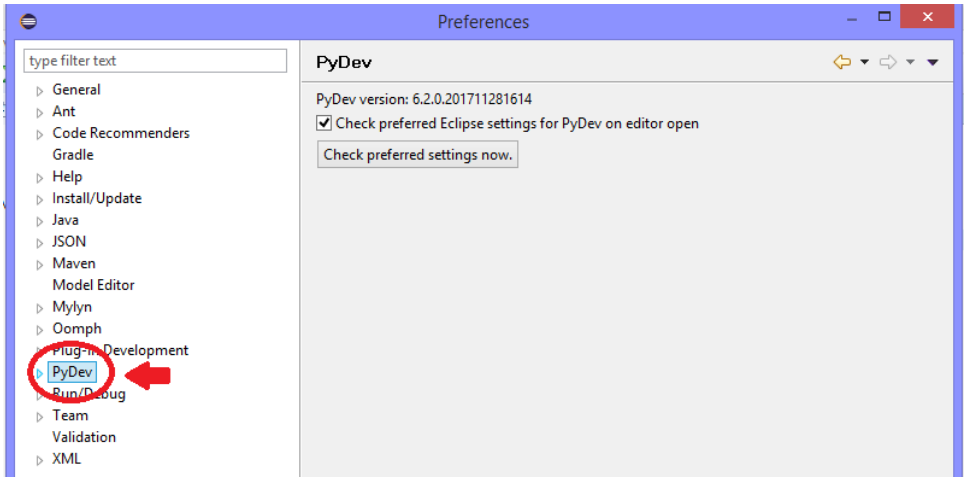

Figura 8: Tercer Paso

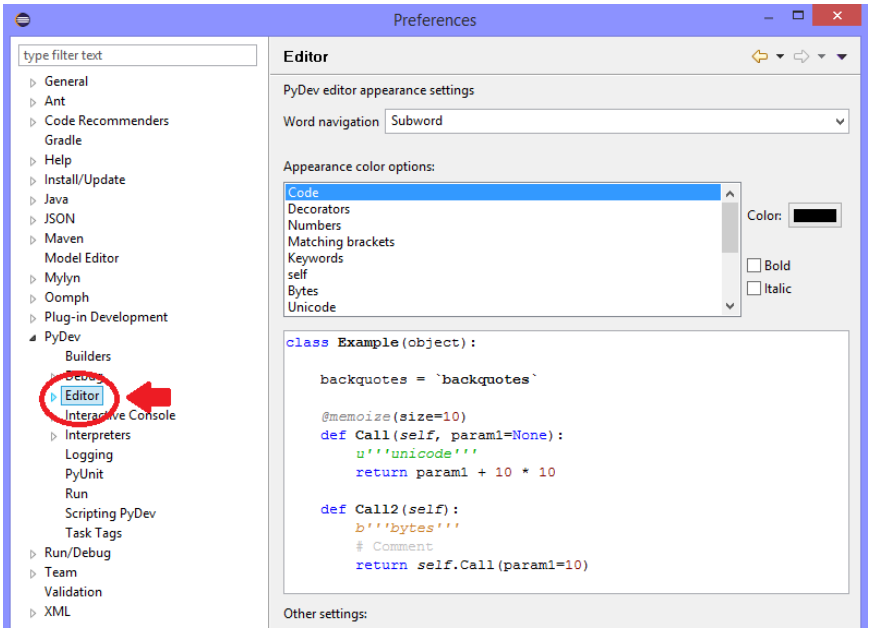

Figura 9: Cuarto Paso

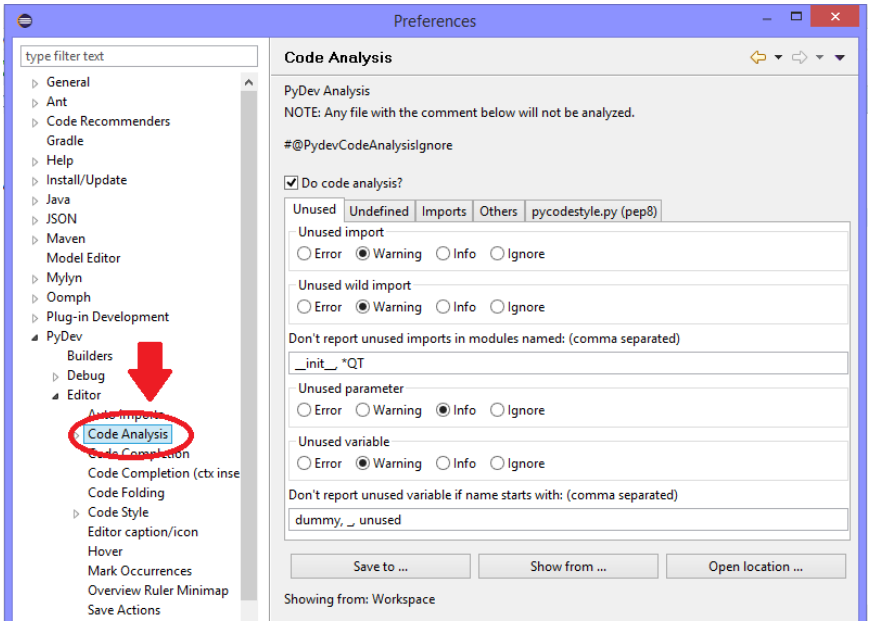

Figura 10: Quinto Paso

Después con la opción de Code Analysis, al lado derecho, se nos habilitarán varias opciones, entre ellas hay una serie de pestañas **(Unused, Undefined, Imports, Others, pycodestyle.py(pep8))**, de las cuales daremos click en la última que es **pycodestyle.py(pep8)**.

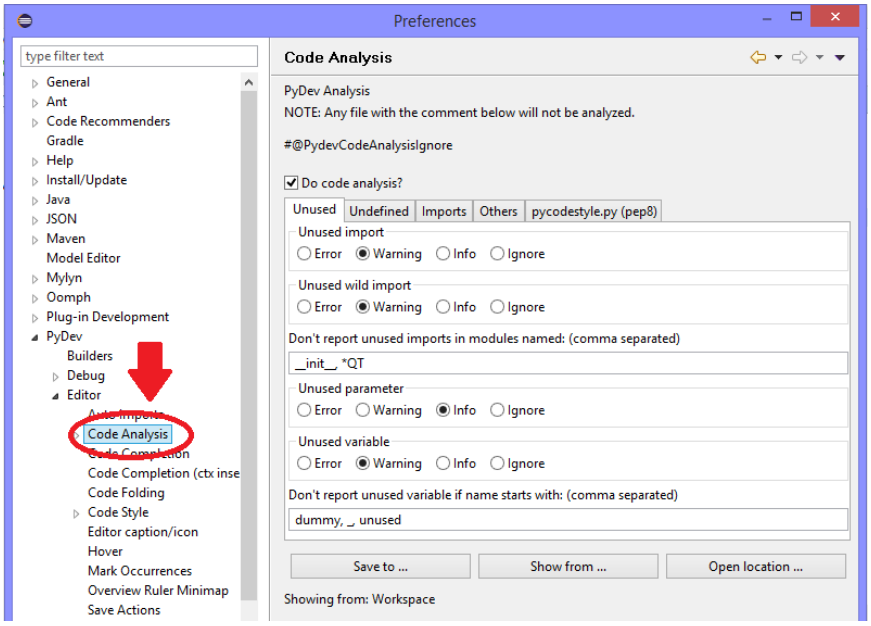

Figura 11: Sexto Paso

Esta última opción posee cuatro tipos de como quiere que se maneje en la codificación **(Error, Warning, Info, Don't run)**, de los cuales para activar la funcionalidad se pueden escoger cualquiera de los primeros tres, recomendamos alguno de los dos primeros, para mejores resultados. Luego de escoger una de la opciones, para terminar el proceso solo presionamos en la opción de **Apply and Close**.

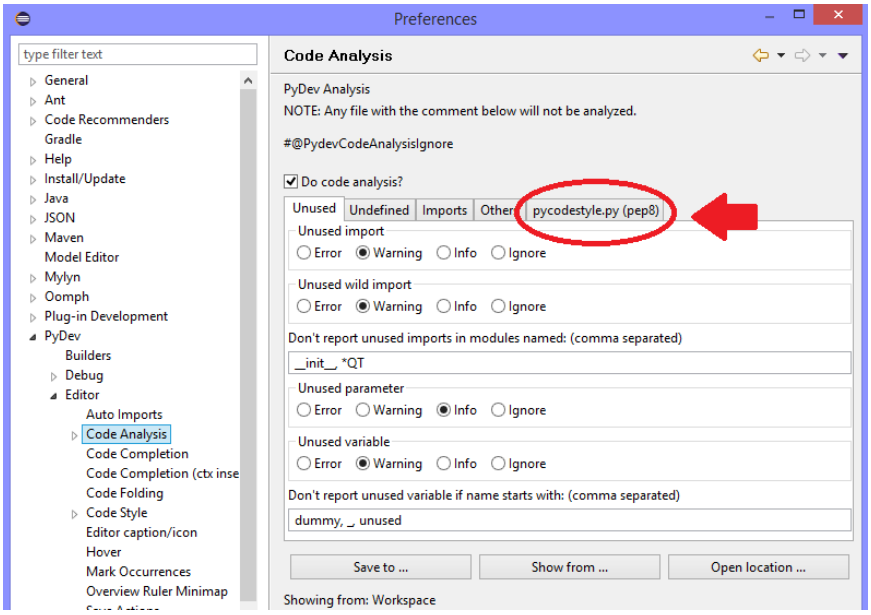

Figura 12: Sétimo Paso

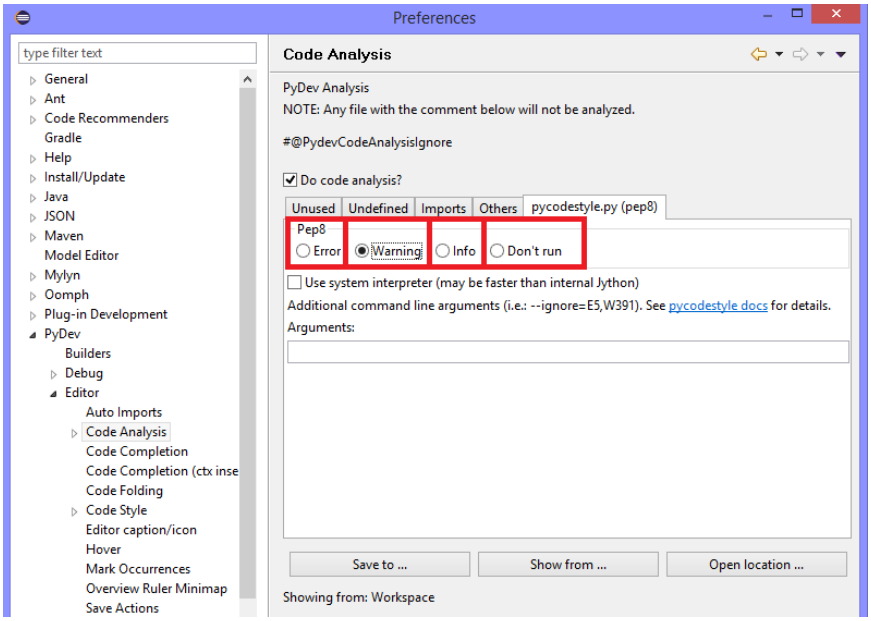

Figura 13: Octavo Paso

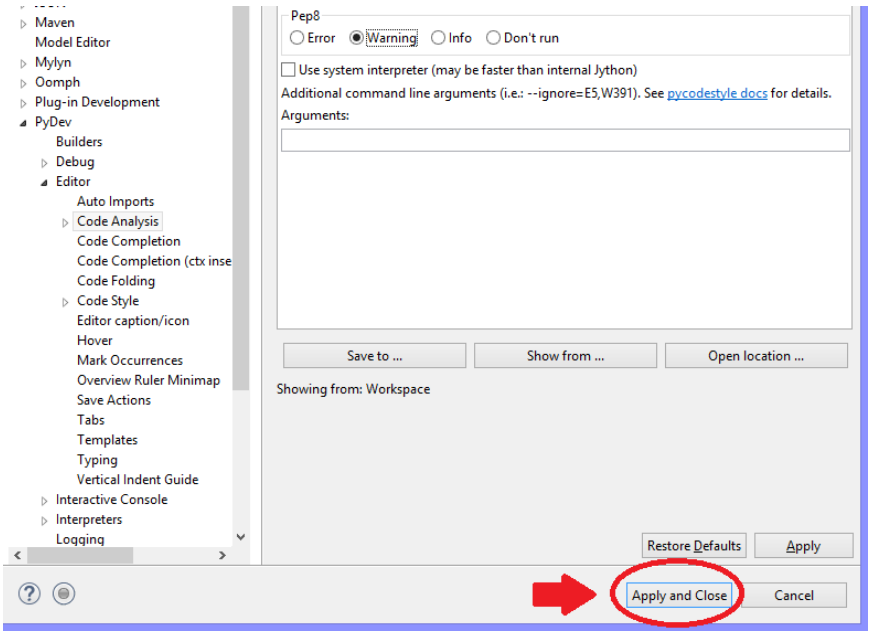

Figura 14: Noveno Paso

6. Herramienta Jenkins

Primero que todo, se debe instalar la herramienta Jenkins en el ordenador. En el siguiente link se puede descargar la versión que el usuario desee, dependiendo de su sistema operativo y cuál versión desea tener: Descarga Jenkins, luego este se debe de instalar de forma correcta con una serie de pasos, los cuales se pueden ver y realizar con el siguiente vídeo Instalar Jenkins, se debe seguir pasos por paso y se instalará de forma correcta (En caso de tener Java ya instalado en el ordenador, obviar dicho proceso). Siempre se debe tener acceso a internet, ya que las acciones que se realicen en Jenkins es desde un navegador web.

La integración de la herramienta Jenkins con Git se debe hacer de la siguiente manera:

La herramienta se abrirá en un navegador web, con un panel de control. Se debe de estar seguro del repositorio que se relacionar con el Jenkins. Luego en la herramienta lo primero que haremos es irnos a la opción de **Administrar Jenkins**, luego de las opciones que nos muestran, escogemos la que se llama **Administrar Plugins**.

Figura 15: Administrar Jenkins

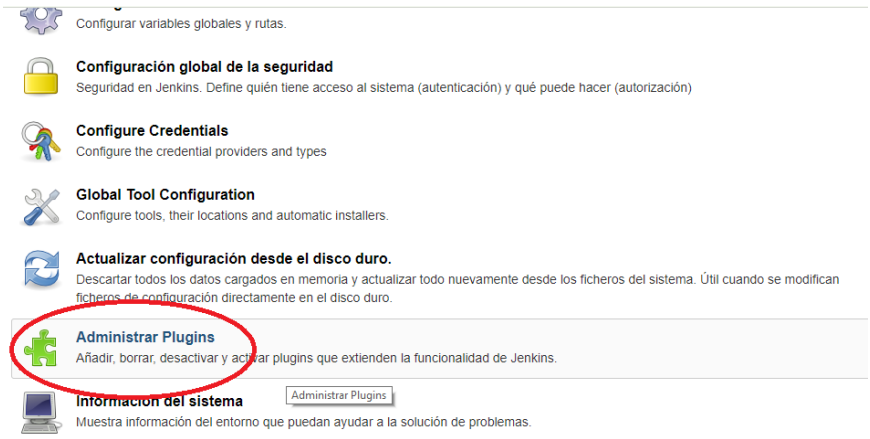

Figura 16: Administrar Plugins

Después de las opciones que nos muestra en la nueva ventana, damos click en la opción de **Todos los plugins**, para luego buscar en en **Filtrar** un plugin para un repositorio (En nuestro caso sería uno de GitHub). De las opciones que nos muestra sobre GitHub, buscamos y escogemos la opción llamada **Github Integration**, para luego dar click en el botón de **Descargar ahora e instalar después de reiniciar**. Esperamos a que descargue y luego reiniciamos Jenkins.

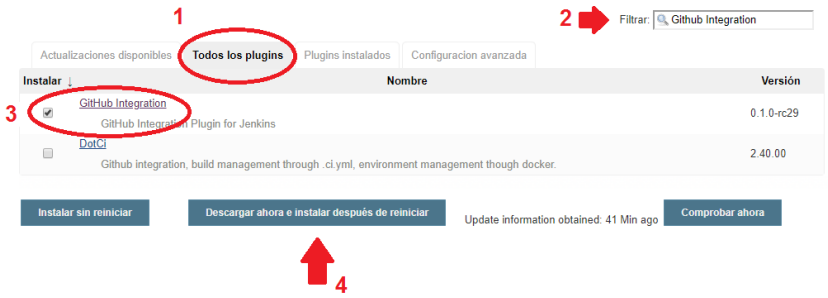

Figura 17: Buscar Plugin de Github

Luego de un rato el plugin se instalará. (Para explicar el ejemplo estamos utilizando un repositorio de prueba, pero el proceso es el mismo para cualquier repositorio). Después, de nuevo en la página principal, damos click en la opción de **Nueva Tarea**.

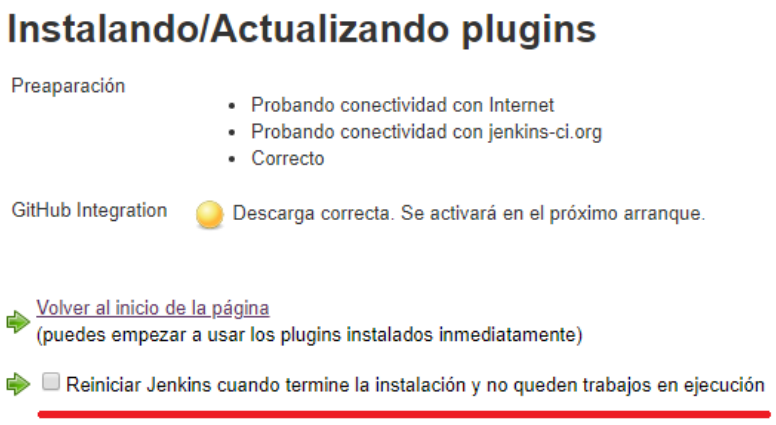

Figura 18: Reinicio de una vez cargado

Figura 19: Nueva Tarea

Nos mostrará una nueva ventana en donde poner el nombre de la nueva tarea (Puede llamarse igual que el repositorio). Luego se escoge una de las opciones de tipos de proyectos, que se pueden realizar, los más comunes son **Crear un proyecto de estilo libre** y **Pipeline**. Nosotros nos inclinaremos por la opción de crear un proyecto de estilo libre en esta ocasión. Escogemos la opción de tipo de proyecto que queremos y damos al botón de **OK**.

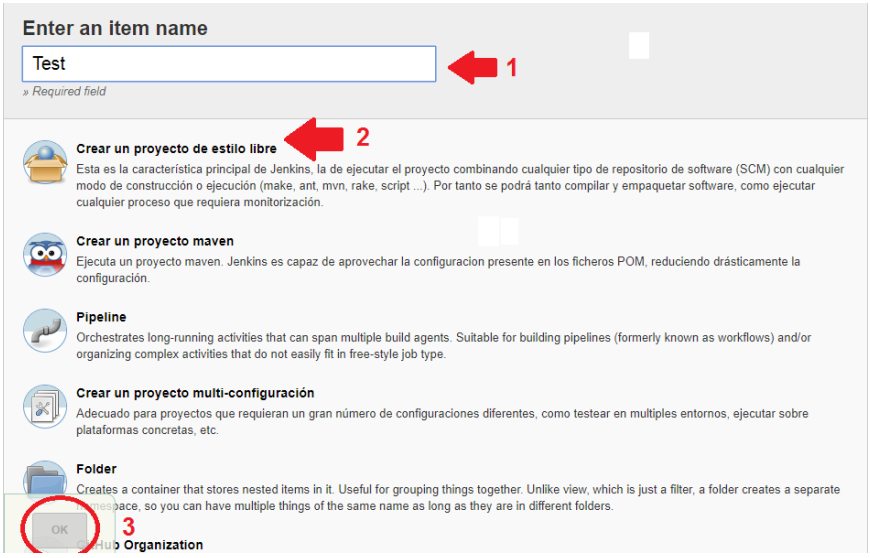

Figura 20: Creación de nuevo proyecto

Este proceso nos enseñará otra ventana totalmente nueva, con varias op-

ciones. En esta, en la sección **General**, seleccionamos la opción de **GitHub Project**, la cual nos habilitará un campo para ingresar el URL de dicho proyecto, por lo cual iremos a nuestro repositorio, copiaremos el link y lo pegaremos en el espacio habilitado

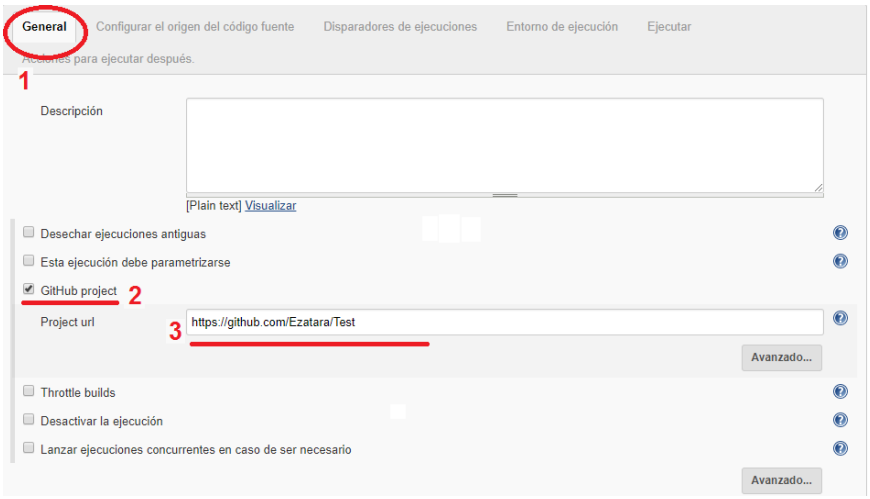

Figura 21: Indicar repositorio

Luego más abajo, en la sección de **Configurar el origen del código fuente**, indicaremos que la opción que usaremos es **Git**, la cual nos habilitará una opción para ingresar un URL del repositorio, pero este se refiere para poder clonar dicho repositorio. Por lo que iremos a nuestro repositorio, que tiene un botón al lado derecho que dice **Clone or download**, damos click en este, copiamos el link y lo pegamos en la opción que nos da Jenkins (Se pueden especificar los branches que se quiere, queda a opción del usuario).

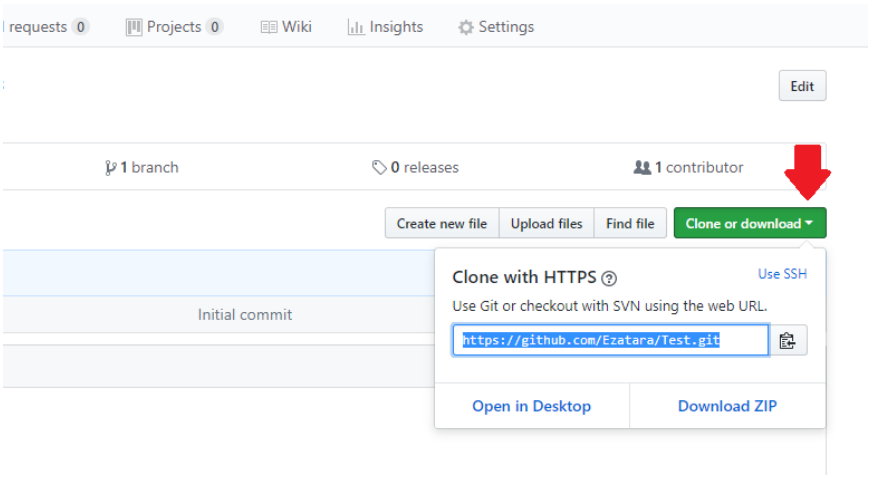

Figura 22: Link para la clonación

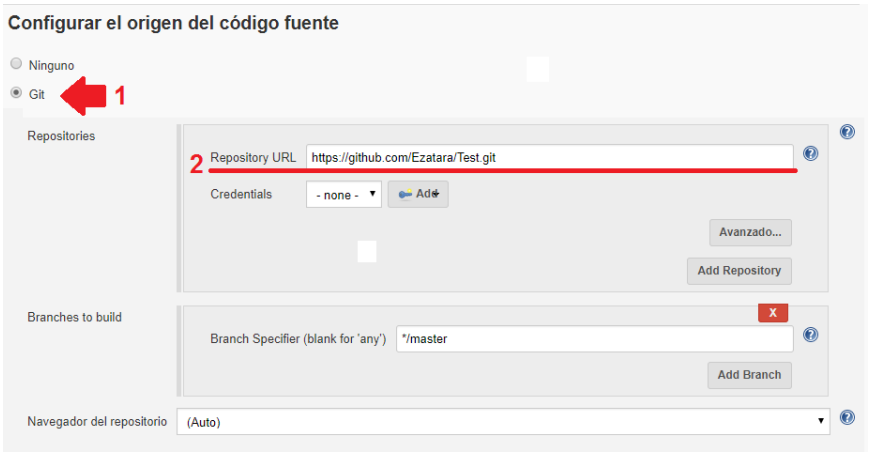

Figura 23: Configuración del origen

En la sección de **Disparadores de ejecuciones**, escogeremos una opción para que Jenkins sepa en qué momento empezar a construir y hacer su trabajo. Escogeremos la opción **Github hook trigger for GITScm polling**, con esto cada vez que esté sucediendo algo en GitHub este se activará. También se puede configurar las opciones de "Entorno de ejecución" o "Ejecutar", en caso de que el usuario lo quiera o lo necesite. Luego de estar satisfecho con estos procesos, damos click al botón de **Guardar** que se encuentra al final.

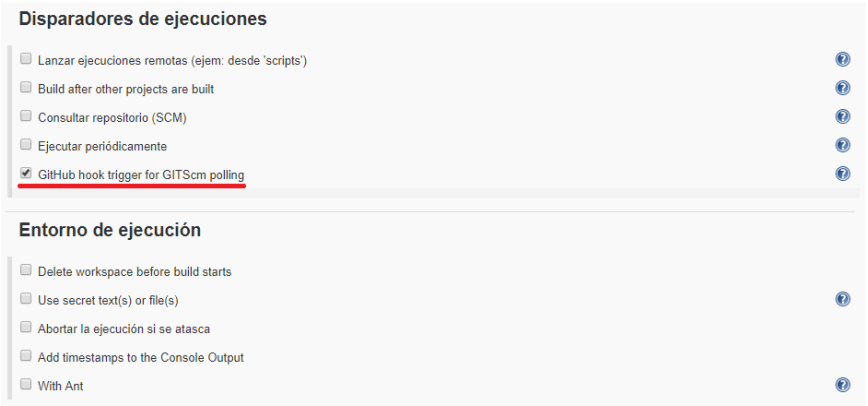

Figura 24: Disparador de ejecuciones

Por último lo que debemos hacer, es configurar GitHub para que se conecte y notifique a Jenkins, cada vez que ocurre o realizamos un commit. Para esto nos vamos de nuevo a nuestro repositorio, nos dirigimos a la opción de **Settings**, luego al lado derecho damos click en **Webhooks** para luego presionar el botón de **Add Webhook**.

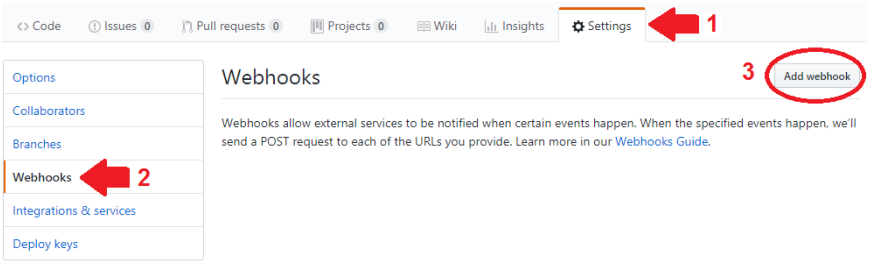

Figura 25: Setting en GitHub para Webhooks

Se nos presentarán varias opciones para adherir un nuevo Webhook, en el espacio de de **Payload URL**, ingresamos la dirección de nuestro Jenkins (Solo los numeros del link) y le adherimos a dicho link **/github-webhook/**. Se puede escoger el tipo de contenido (Content type) si así el usuario lo desea. Luego escogemos en que caso queremos que el webhook se active (A gusto del usuario) y finalmente presionamos el botón **Add webhook**. Debe aparecer un nuevo link entre la opciones de Webhooks.

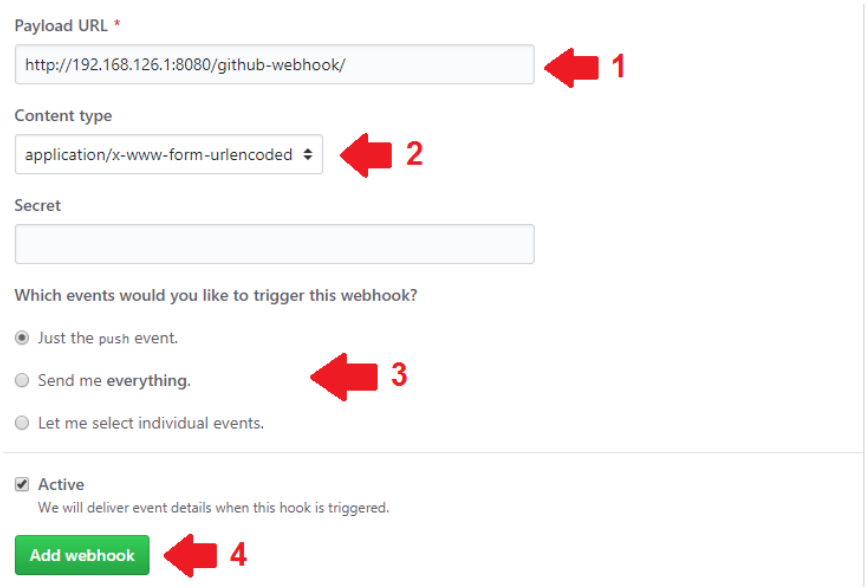

Figura 26: Adhición de nuevo Webhook

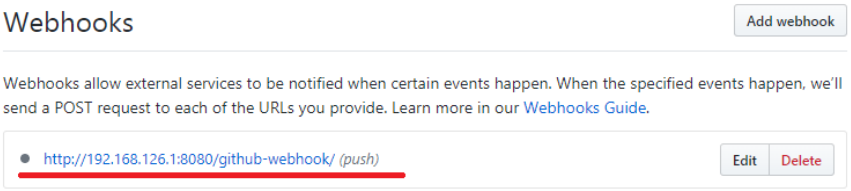

Figura 27: Éxito de integración

Así estaría integrado la herramienta Jenkins con Git, la cual en hacer algún commit en algún documento que posea dicho repositorio se verá reflejado en Jenkins y se podrá condicionar la codificación y documentación.

7. Cuantificación de Métricas

Se va a utilizar GrimoireLab para la cuantificación de métricas. para instalar seguir los siguientes pasos:

- Ejecutar en el Shell pip install grimoire-mordred

8. Pruebas unitarias

- Cargar imagen
  - Entrada esperada: imagen PNG.
  - Entrada atípica: Cualquier otro archivo.
  - Salida esperada: Edad estimada.
- Ejecución archivo Python.
  - Entrada esperada: archivo .py
  - Entrada atípica: Cualquier otro archivo.
  - Salida esperada: ejecución de script exitoso.
- Leer datos del paciente
  - Entrada esperada: String.
  - Entrada atípica: Números.
  - Salida esperada: Lectura exitosa.
- Guardar datos
  - Entrada esperada: String.
  - Entrada atípica: int, campo en blanco.
  - Salida esperada: Archivo guardado.