Aseguramiento de la Calidad del Software: Avance 2, proyecto semestral

Yin Cheng Liang 2015018200
Jose Antonio Salas 2015013633
Emanuel Alvarado 2015183027

16 de Octubre de 2018
Git: https://github.com/joseant12/CellSegmentationProject

1 Introduccion

En el presente avance se desarrollarán las funcionalidades críticas del proyecto, usando criterios de aseguramientode calidad. En el documento se detallan varios requerimientos fijados por el cliente.

2 Documento SyRS actualizado

2.1 Propósito del sistema

El desarrollo de este sistema tiene el propósito de agilizar y facilitar el proceso de investigación en la industria biomédica por medio del análisis automatizado de imágenes biomédicas obtenidas con la ayuda de microscopios electrónicos de fluorescencia. El análisis manual de miles de imágenes es una tarea difícil y agotador para un experto, por eso con la ayuda del sistema se podrá ahorrar una cantidad importante de tiempo y esfuerzo para utilizarla para realizar otras actividades. Además de poder generar con el sistema resultados eficientes a gran escala.

2.2 Alcance del sistema

El sistema de análisis automatizado de células tendrá la funcionalidad de procesamiento de imágenes biomédicas que consiste en la segmentación y conteo de las mismas dado una serie de imágenes provistas por los doctores. Al finalizar el proceso se generan varios archivos con el informe del análisis detallando la cantidad de células y la posición de cada una en cada cuadro.

El sistema no será para todo público, sino solamente para los doctores y personal autorizado del Laboratorio de Quimiosensibilidad tumoral de la Universidad de Costa Rica.

Los beneficios al utilizar este sistema es la mejora en el tiempo de respuesta del análisis, disminuyendo el esfuerzo por parte de los expertos y así agilizando el proceso de atención a los pacientes.

2.3 Resumen del sistema

2.3.1 Contexto del sistema

El sistema funcionara en una aplicación web en donde se tendrá acceso a una o varias direcciones provistas por el usuario en donde se encuentran localizadas las imágenes biomédicas a analizar.

El usuario dará una dirección, a partir del cual se cargan las imágenes obtenidas usando microscopios electrónicos de fluorescencia a la aplicación. Luego se procederá al proceso de análisis de las imágenes y al finalizar el usuario podrá visualizar el informe generado de los resultados del estudio en pantalla y en archivos aparte.

El diagrama de la arquitectura del sistema se muestra en la Figura **??**.

2.3.2 Funciones del sistema

El sistema tendrá la capacidad de cargar una serie de imágenes biomédicas en escala de grises adquiridas mediante un microscopio electrónico y procesarlas de manera que facilite su análisis para luego hacer la detección y el conteo de objetos importantes mostradas en la imagen. En la etapa final del proceso, además de mostrar los resultados obtenidos del análisis en pantalla, se generarán archivos de imágenes y archivos de informes en formato .csv.

2.3.3 Características del usuario

Los usuarios finales, los cuales utilizaran la aplicación serán microbiólogos con conocimientos generales de herramientas informáticas y que forman parte del equipo del Laboratorio de Quimiosensibilidad Tumoral de la Universidad de Costa Rica.

2.4 Requerimientos funcionales

2.4.1 Cargar secuencia de imágenes

Dado una dirección provista por el usuario se podrá cargar un conjunto de imágenes biomédicas tomadas con microscopios electrónicos de fluorescencia.

2.4.2 Detección y conteo de objetos

Después de haberse cargado las imágenes, con esta funcionalidad se lograra detectar y contar los objetos de interés de las imágenes como son las células, y mostrarlas en la interfaz gráfica para su visualización.

2.4.3 Generar informe

Al finalizar el proceso de análisis, los resultados obtenidos se guardarán en archivos en formato .csv en donde se indicará la cantidad de células, la posición de cada una en cada cuadro, entre otras características de los objetos encontrados.

2.4.4 Visualizar proceso

Se mostrará los detalles por cada imagen o por lote de imágenes en pantalla del proceso de análisis en cuando esta finalice. Como por ejemplo visualizar el tiempo de ejecución.

2.4.5 Cálculo de métricas

A partir de una imagen se utilizara una métrica, por ejemplo la métrica de Dice, para cuantificar la precisión de la segmentación, evaluando la diferencia de señales binarias y luego generar una imagen ground truth con los blobs de los objetos de interés manualmente marcados.

2.5 Requerimientos de usabilidad

El sistema debe de tener facilidad de aprendizaje y usabilidad para los usuarios ya que como es una aplicación de análisis de imágenes biomédicas, esto requiere un tiempo estimado en su ejecución, y mientras menor sea el tiempo invertido en aprender cómo funciona y usarlo mejor. Y en cuanto a la interfaz gráfica, debe de ser simple y cómodo para el usuario, de manera que los microbiólogos puedan entender el sistema y encontrar las funcionalidades necesarias para su trabajo.

A continuación se muestran los requerimientos de usabilidad del sistema de segmentación de células

1. Requerimiento 1:Fácil de aprender. los usuario no deben toman más de 5 minutos en aprender las funcionalidades del sistema.

2. Requerimiento 2:Fácil de usar. los usuario no deben durar más de 5 minutos. aprendiendo como a como usar el sistema.

3. Requerimiento 3:Funcionamiento de la interfaz gráfica.Debe ser simple y cómoda para el usuario, de manera que se pueda encontrar las funcionalidades fácilmente.

2.6 Requerimientos del rendimiento

La apertura de la aplicación y la carga de todas las funcionalidades en la interfaz gráfica no debería de durar mucho ya que es un sistema simple es esta parte. La carga de imágenes junto con su procesamiento y la generación de informes podría durar su tiempo, ya que dependiendo de la cantidad de imágenes a

analizar podría disminuir o aumentar el tiempo de ejecución pero lo ideal es que se trate de durar lo menos posible y así agilizar el proceso de investigación.

A continuación se muestran los requerimientos de rendimiento del sistema de segmentación de células:

1. Requerimiento 1:El tiempo de acceso a la aplicación no debe durar más de 10 segundo.

2. Requerimiento 2:El tiempo máximo para la carga y procesado de una o más imágenes dede estar entre 10 segundos y 2 minutos.

2.7 Interfaces del sistema

El sistema se base en la arquitectura Cliente-Servidor por lo que dispondrá tanto de una interface para cliente y para el servidor. Se dispondrá de una interfaz gráfica en la cual el usuario pueda realizar todas las funcionalidades que el sistema va a disponer para los usuarios finales.

2.8 Operaciones del sistema

2.8.1 Requerimientos del integración del sistema humano

El sistema debe ser altamente tolerante a fallos provocados por la interacción con el usuario, con lo cual se espera que el sistema al tener problemas por la mala manipulación del mismo sea capaz en la mayor parte de poder recuperarse y así poder seguir funcionando correctamente, esto para poder procesar la mayor cantidad de imágenes posibles.

El sistema al procesar imágenes puede darse el caso en que el usuario ingrese imágenes incorrectas o que del todo no se puedan procesar, pudiendo provocar errores en el sistema.

2.8.2 Mantenibilidad

Al esperarse poder procesar una secuencia de imágenes que puede ser en algunos casos de una cantidad considerable, puede darse los casos que el sistema se sature o bloque lo cual implicaría una revisión y mejora en el código fuente para optimizar el tiempo de procesamiento. Se espera que el tiempo de mantenibilidad no sobrepase más de 5 horas.

2.8.3 Confiabilidad

El usuario final del sistema son médicos y especialista en la interpretación de imágenes médicas, los cuales darán al sistemas las imágenes correctas para su posterior procesamiento, al ser dirigido para estos usuario se logra un nivel altamente positivo de confiabilidad.

2.9 Modos y estados del sistema

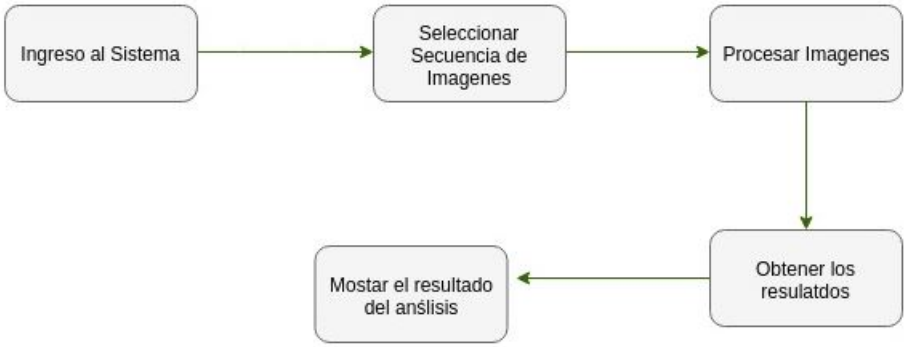


Figure 1: Modos y estados del sistema

2.10 Características físicas

2.10.1 Requerimientos físicos

El sistema va ser instalado en equipos propios de los especialistas médicos, en el laboratorio de trabajo de ellos mismo, aparte del computador solo se va a necesitar un monitor donde se pueda realizar las tareas del sistema.y muestre resultados.

2.10.2 Requerimientos de adaptabilidad

En caso de que el volumen de procesamiento de imágenes crezca rápidamente se debe de necesitar de mayor poder de procesamiento lo que implicaría en un futuro, la adquisición de más módulos de memoria ram y procesador.Para que el sistema cumpla con sus tareas en un tiempo considerablemente rápido.

2.11 Condiciones ambientales

Al ser un sistema de procesamiento de imágenes es recomendable que el equipo esté en un lugar fresco, para que la alta temperatura no afecte el funcionamiento del equipo al poder llegar a ser sobrecargado con mucho proceso.

2.12 Seguridad el sistema

El sistema será utilizado por especialista con un gran conocimiento en imágenes médicas para el tratamiento de enfermedades. Con respecto a las imágenes que fueron ya procesadas, serán almacenadas en una base de datos para mantener una copia de seguridad de las mismas, la base de datos estará protegida por su respectivo usuario y contraseña para prevenir posible mala manipulación de las misma.

2.13 Administración de la información

El sistema de análisis de segmentación de células recibe como entrada tres atributos para crear una colección de imágenes. Estos campos corresponden a un título de colección, una descripción de no más de 256 caracteres y por último una o más imágenes. Estas imágenes deben coincidir con las características de los ejemplos. Se recomienda que las imágenes sean formato png, aunque cabe aclarar que el sistema soporta imágenes formato jpg. La salida de datos consta de un archivo en formato csv donde mostrará el centroide y el área de cada objeto encontrado por el sistema.

2.14 Regulaciones y políticas

En el apartado de políticas del sistema, se encuentra la confidencialidad con los datos suministrados por el cliente. Mediante la funcionalidad de registro/login los clientes acceden a su cuenta privada y suben su colección de imágenes a evaluar, para luego recolectar los resultados correspondientes, sin que ningún agente externo interfiera en el proceso. El sistema no se hace cargo de irregularidades en los resultados propuestos, por ello se insta al usuario a verificar la correctitud de los datos.

2.15 Sostenibilidad del ciclo de vida del sistema

El sistema será sometido a pruebas rigurosas para asegurar una calidad óptima en pro de los usuarios. Para ello se tiene control de una serie de procedimientos a efectuarse luego de cada sprint en el transcurso de desarrollo de la aplicación. También se tiene programado realizar mediciones al software con respecto a una serie de métricas previamente establecidas, con el objetivo de tener procesos de control de calidad exitosos. El sistema cuenta con una documentación robusta que garantiza su mantenibilidad. Se espera capacitar al usuario por medio de guías y documentación con tal de satisfacer sus necesidades.

2.16 Empaquetado, manejo, envío y transporte

El sistema consta con atributos de calidad como instalabilidad y adaptabilidad, ya que consta de archivos digitales que pueden ser instalados en cualquier computador con soporte de Django. El sistema está contenido en un repositorio en línea lo cual lo hace fácilmente accesible.

2.17 Verificación

La verificación del sistema está a cargo de agentes tanto internos como externos. Se utilizará herramientas como Sonarqube, Reqview, PyUnit, Doxygen, análisis de salidas, f1 score y entrevista con clientes. Para mayor información al respecto, se recomienda ver el documento de "Atributos de calidad y métricas" (correspondiente a la tarea 1) donde se expone detalladamente el uso de cada una de estas herramientas.

2.18 Suposiciones y dependencias

En cuanto a las suposiciones y dependencias se anuncia lo siguiente:

1. El usuario tiene un conocimiento mínimo del campo biológico correspondiente a las funciones del sistema.
2. El usuario tiene el conocimiento mínimo en cuanto al uso de una computadora y acceso a la web.
3. Las imágenes suministradas al sistema son únicamente de células con el formato correcto.
4. El usuario está consciente que las imágenes suministradas al sistema pueden ser manipuladas como este desee.

2.19 Priorización

1. Usabilidad 2.
2. Funcional 1.
3. Funcional 2.
4. Rendimiento 1.
5. Rendimiento 2.
6. Funcional 3.
7. Usabilidad 1.
8. Usabilidad 3.
9. Funcional 4.
10. Funcional 5.

3 Validación de diseño

Item de Diseño	Requerimiento
Segmentador de Células Web	Funcional-Requerimiento 2 Funcional-Requerimiento 1 y Usabilidad-Requerimiento 1,2,3

Table 1: Caption

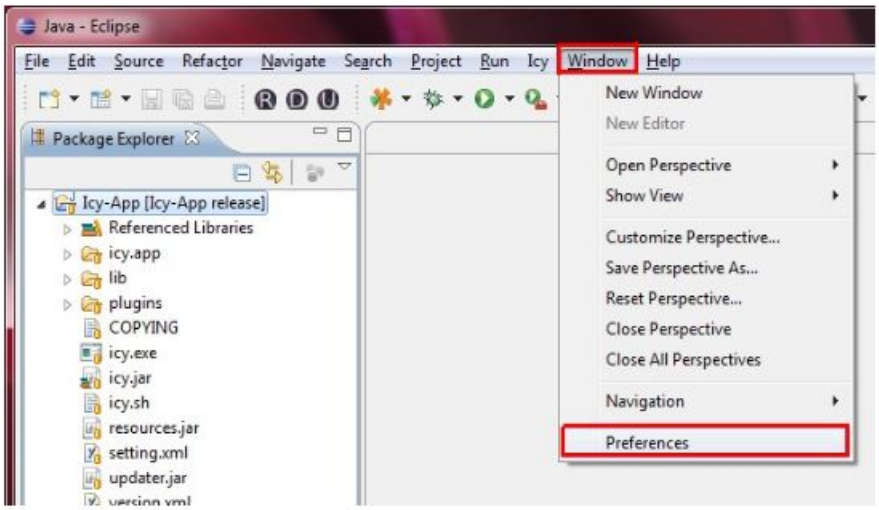
4 Modo de uso de la herramienta de verificación de la codificación y/o documentación

4.1 Verificación de la codificación

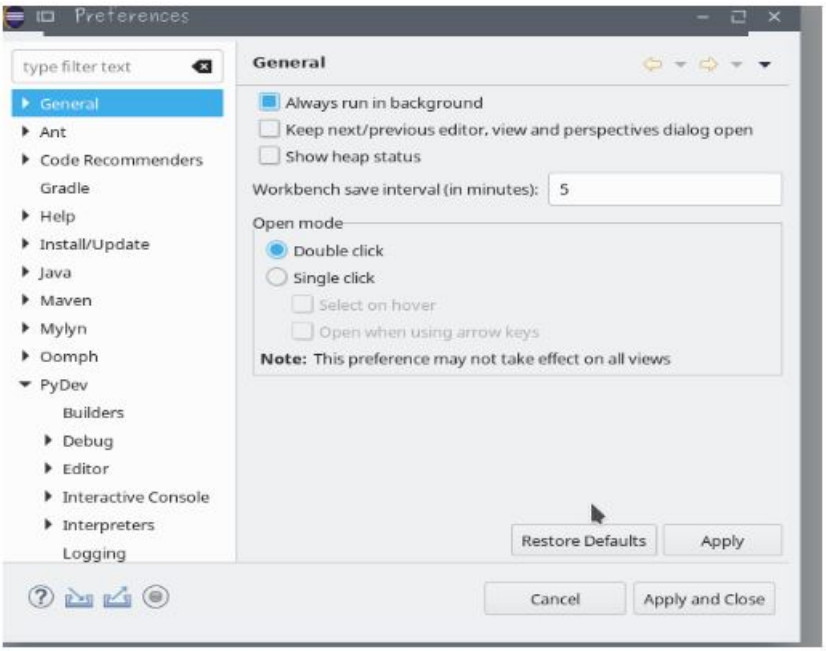
Para el análisis de código se usa la herramienta Code Style para el ide eclipse.

4.1.1 Modo de instalación

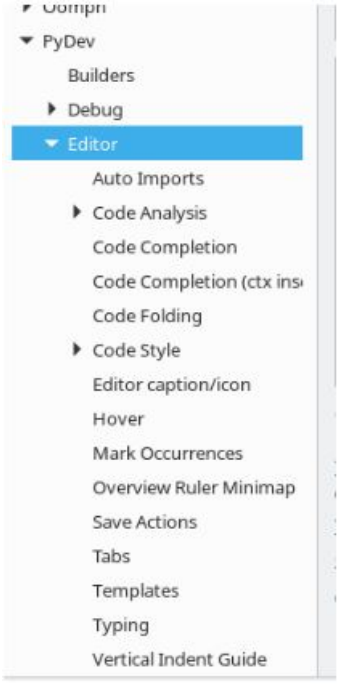
1. Ya dentro del ide nos vamos a la pestaña windows y dentro de ella seleccionamos la subpestaña preferences.



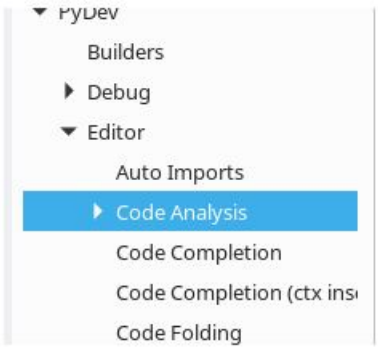
2. Dentro del menú preferences vamos a la opción llamada Pydev.



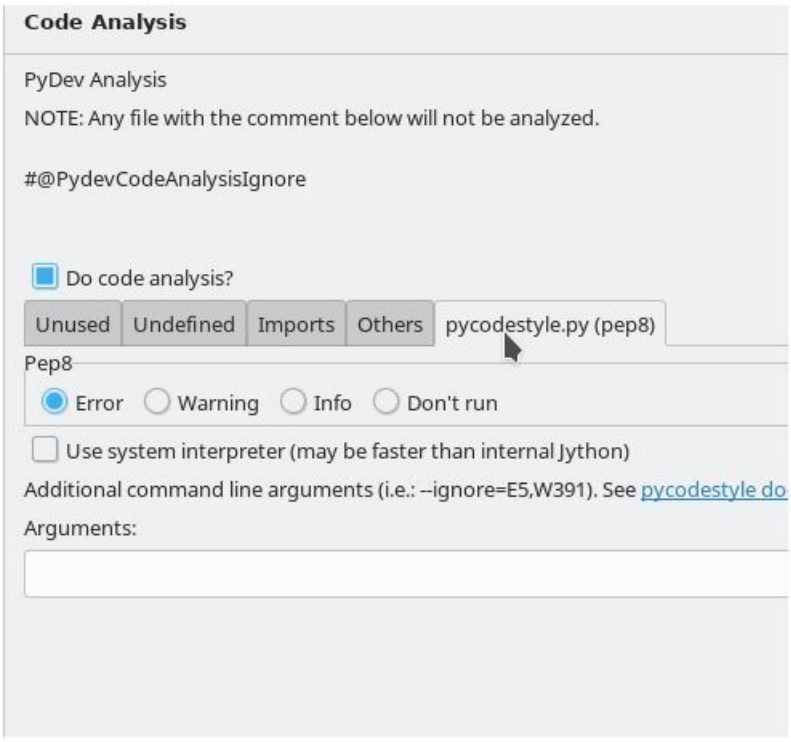
3. Posteriormente damos click sobre la pestaña Editor.



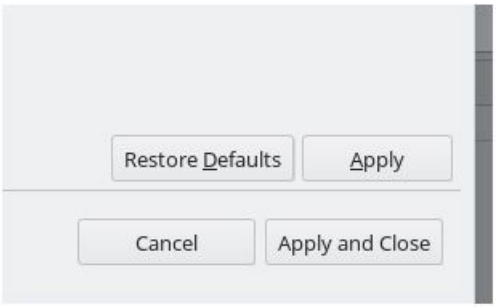
4. Damos click en la pestaña Code Analysis.



5. Nos vamos a la pestaña que tiene el nombre (pep8) y seleccionamos la caja error.



6. Por último damos en el botón apply and close y con esto ya tenemos nuestra herramienta configurada.



4.1.2 Verificación

En caso de que tengamos un error en el estándar de codificación el ide se mostrará de la siguiente forma para el programador.

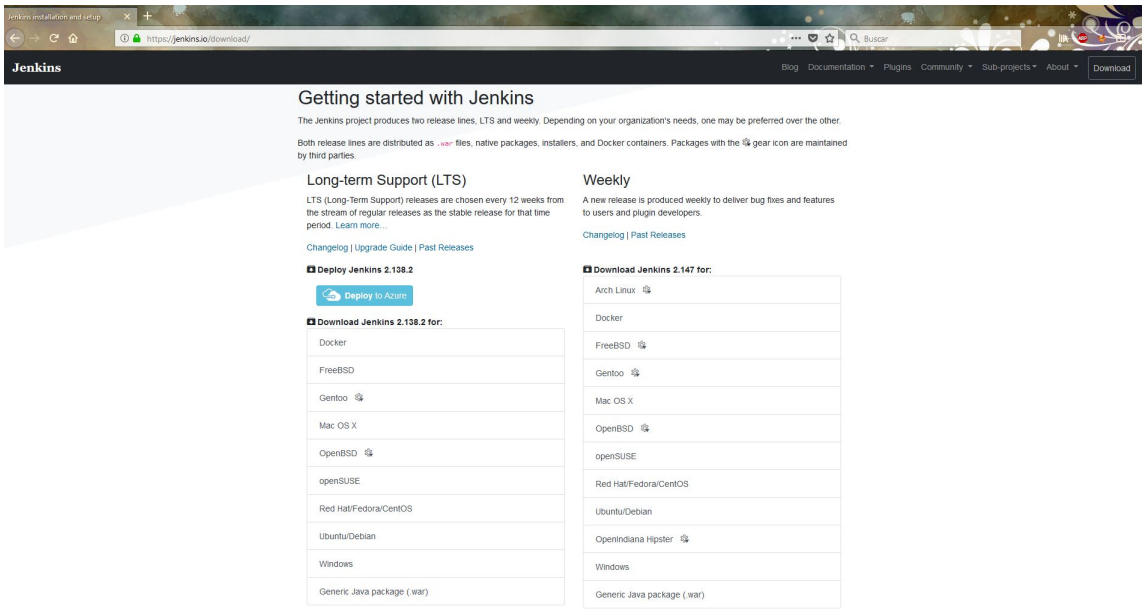
```
81 def preprocess(imgs):
82     imgs_p = np.ndarray((len(imgs), img_rows, img_cols), dtype=np.float32)
83     for i in range(len(imgs)):
84         imgs_p[i] = imgs[i].reshape((img_rows,img_cols))/255.
```

Donde el la linea 84 marca error en el formato de codificación pep8. por supuesto esto no significa que el programa no se ejecutará, aquí solo nos indica que la forma en que escribimos el código no está de acuerdo a pep8.

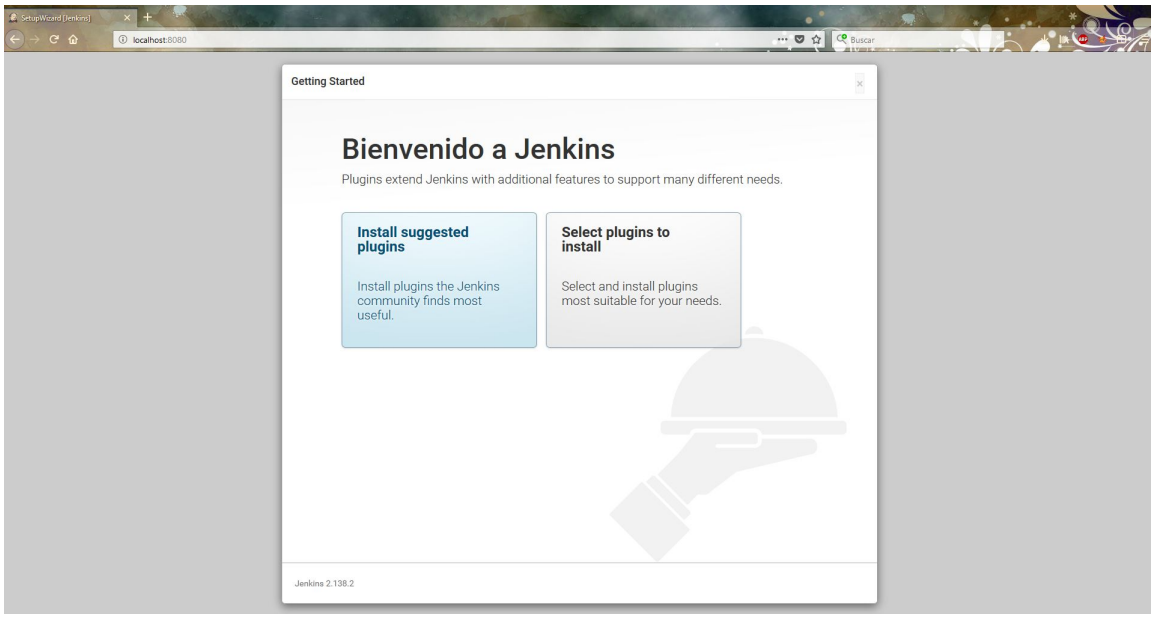
4.2 Uso de la herramienta Jenkins

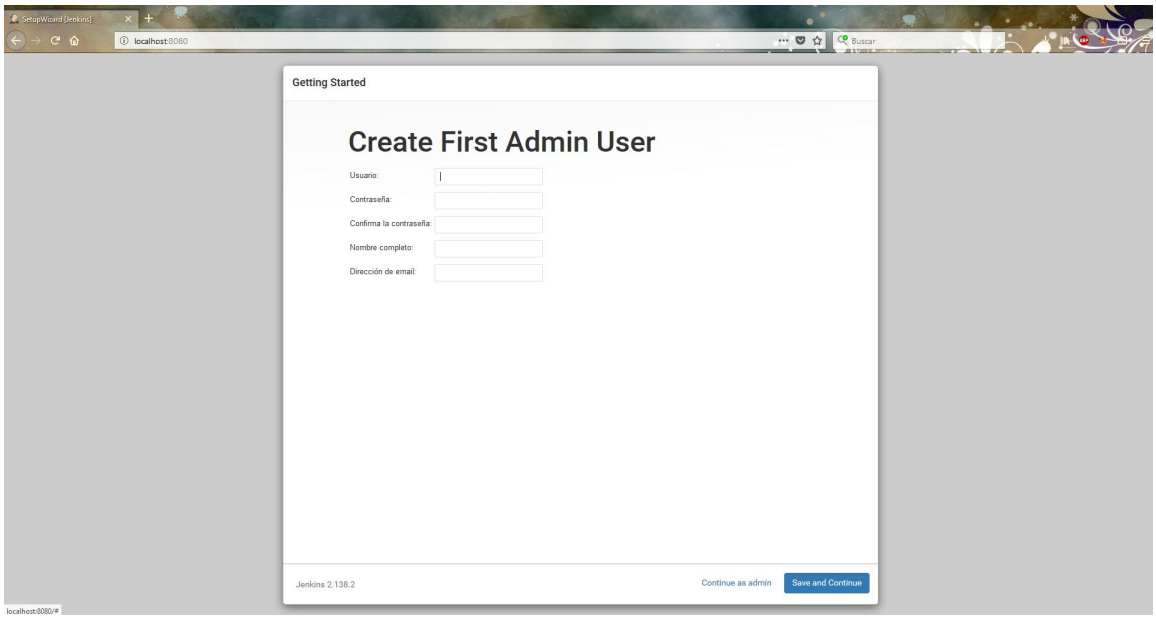
Junto con git se ha usado la herramienta llamada Jenkins para condicionar la modificación del repositorio a la verificación de la codificación y documentación. A continuación se muestran los pasos para utilizar la herramienta:

Para la descarga e instalación de la herramienta hay que ir a la pagina https://jenkins.io/download/ y seleccionar la version que se desea utilizar, en este caso se descargo la versión LTS de Windows. Se descomprime el archivo descargado, se ejecuta el instalador:

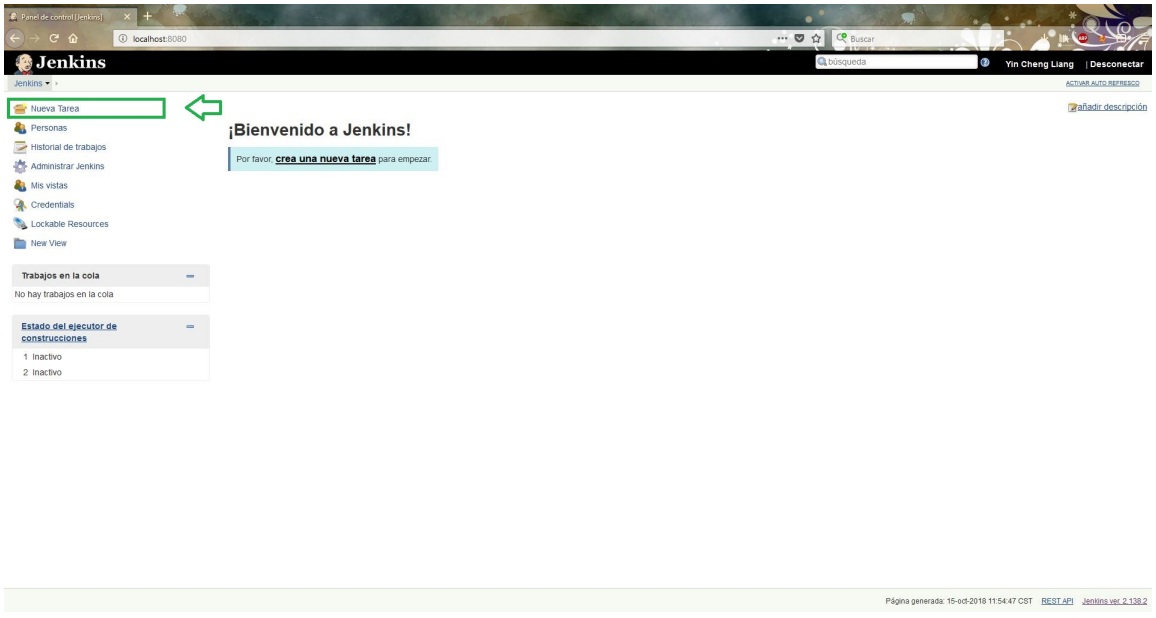


Al terminar la instalación se abrirá el navegador web con la dirección http://localhost:8080/, en donde se procederá a instalar algunas funcionalidades necesarias para el uso de la herramienta y luego la creación de un usuario administrador:

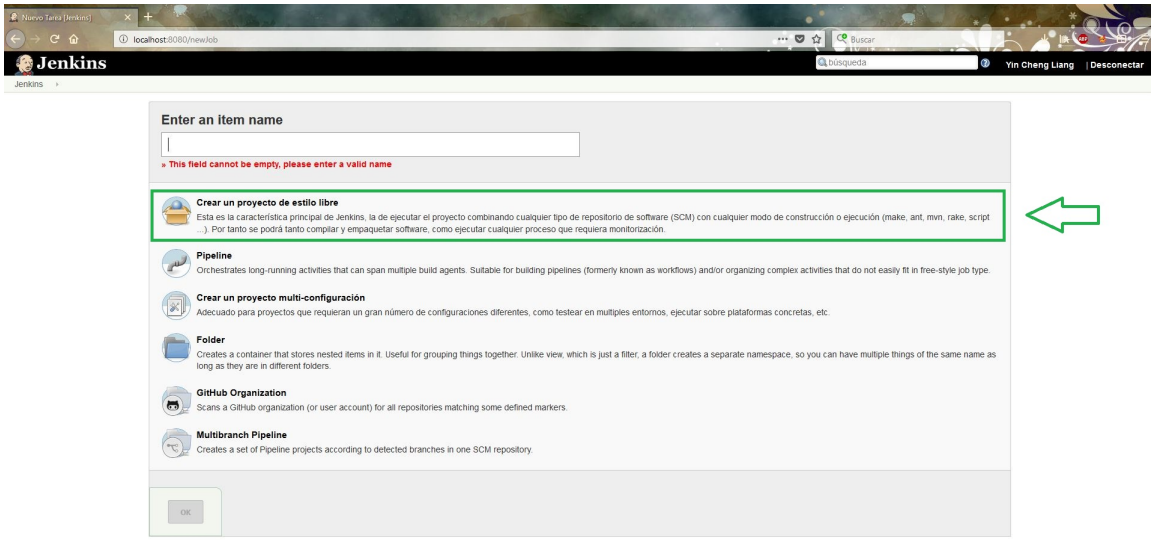




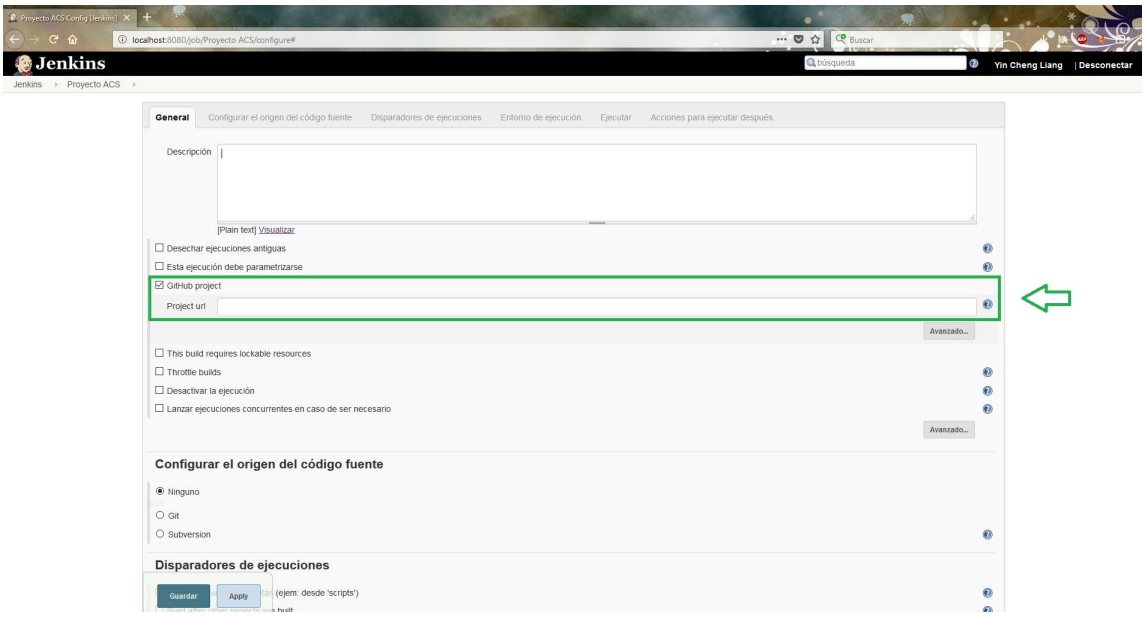
Cuando se entra a la página de la herramienta se podrá observar algo similar a la imagen mostrada a continuación, y para empezar a utilizarla se procede a crear una nueva tarea:



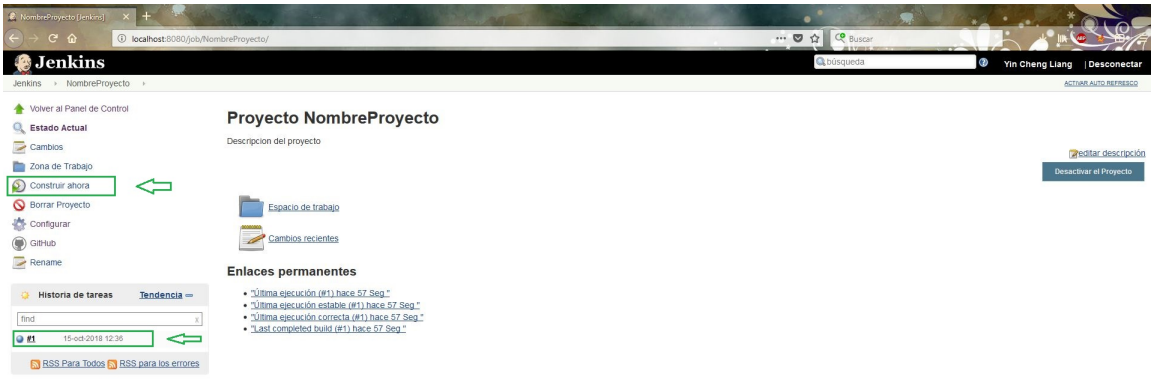
Se digita un nombre para el proyecto a desarrollar y se elige la opción del tipo de proyecto que se desea, en este caso se elige proyecto de estilo libre:



Luego se procese de llenar los datos de configuración del proyecto, en donde se podrá agregar el url del proyecto de github, y para finalizar se da click al botón de aplicar y guardar:



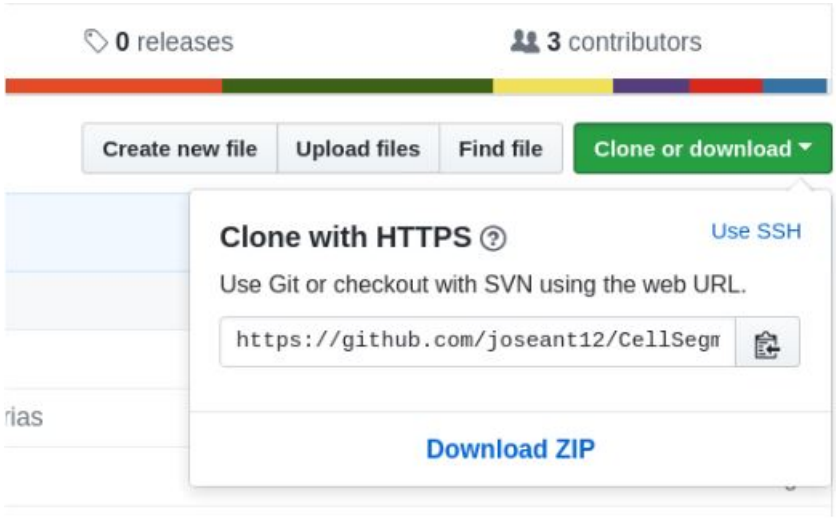
Y para finalizar se revisa la información proporcionada, luego se da click en "Construir ahora" para empezar la verificación del proyecto y en la parte inferior izquierda de Historia de tareas aparecerá un punto de color azul que significa que todo está correcto en el proyecto y un punto rojo si es el caso contrario:



5 Obtencion de la versión actual del sistema

Para obtener la última versión del proyecto se debe ingresar a la siguiente dirección web: https://github.com/joseant12/CellSegmentationProject.git

Ya cuando ingresemos a esta dirección debemos ir al botón llamado clone or download, y seleccionar download como zip, luego procederá a descargarse.



Método arternativo: Para este método se utiliza la consola, se utiliza el comando clone seguido de la dirección web del repositorio.

```
git clone https://github.com/joseant12/CellSegmentationProject.git
Clonar en «CellSegmentationProject»...
remote: Enumerating objects: 761, done.
remote: Counting objects: 100% (761/761), done.
remote: Compressing objects: 100% (427/427), done.
remote: Total 761 (delta 451), reused 619 (delta 312), pack-reused 0
Receiving objects: 100% (761/761), 117.50 MiB | 6.92 MiB/s, done.
Resolving deltas: 100% (451/451), done.
```

6 Cuantificacion de metricas

6.1 Primer métrica

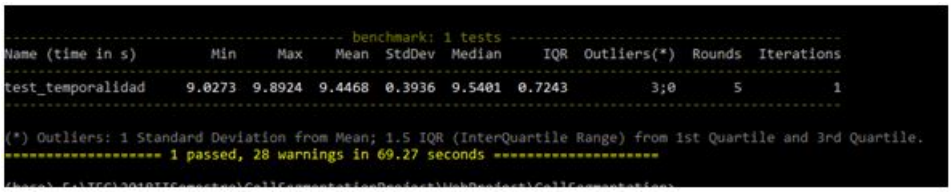
1. Categoría: Portabilidad.
2. Atributo: Adaptabilidad.
3. Métrica: El sistema debe estar disponible en los principales 5 navegadores web.

Sin importar el navegador que se utilice, el sistema debe brindar todas las funcionalidades por igual. El sistema utiliza HTML 5 y Bootstrap como herramientas en el frontend, por lo cual debe ser evaluado su disponibilidad y adaptabilidad a los navegadores de uso más frecuentes. El equipo de calidad realizará pruebas de funcionalidad en los navegadores Google Chrome, Mozilla Firefox, Microsoft Edge, Opera y Safari para demostrar la adaptabilidad del sistema sin importar el navegador que el cliente use para acceder.

6.2 Segunda métrica

1. Categoría: Eficiencia.
2. Atributo: Tiempo de respuesta.
3. Métrica: El sistema debe durar máximo 30 segundos procesando un lote de 50 imágenes.

Durante el desarrollo del proyecto se propuso que, para medir la eficiencia del sistema, este debía ser capaz de procesar un lote de al menos 50 imágenes en un tiempo no menor a 30 segundos. Claro está, estas pruebas dependen de los recursos con los que cuente el equipo que vaya a alojar el servidor. Para realizar las correspondientes mediciones, se creó un test en pytest nombrado test_Temporalidad, dentro de la carpeta CellSegmentation del módulo principal del proyecto. Este test se encarga de definir una carpeta con un lote de 50 imágenes y llamar al módulo segmentador de imágenes para procesarlo. Para cuantificar los resultados, utilizamos el plugin externo de benchmark de pytest. Los resultados obtenidos fueron los siguientes:



Luego de desarrollar y ejecutar el test, los resultados fueron contundentes. El test en total duró aproximadamente 69 segundos, pero el módulo de carga de imágenes fue dividido en 5 rondas cuya duración promedio fue de 9.44 segundos, lo cual deja como resultado un tiempo total de 47 segundos.

Dado el nuevo resultado, la métrica que mide la categoría eficiencia con su atributo Tiempo de respuesta debe ser actualizada a: El sistema debe durar máximo 1 minuto procesando un lote de 50 imágenes.

7 Pruebas unitarias

7.1 load_test_data

Entradas esperadas: Dirección de una imagen para procesar.

Entradas atípicas: Archivo que no exista.

Salidas:Una lista formada por un arreglo con los valores de una imagen convertida a escala de grises, junto con un diccionario que contiene una clave numérica o id , junto con el nombre de la imagen a procesar.

7.2 Dice_coef

Entradas esperadas: Conjunto de x, y valores decimales no vacíos o de diferente tipos.

Entradas atípicas :Conjunto de x, y valores decimales no vacíos o de diferente tipos.

Salidas:Valor flotante que indica el coeficiente de dice de los parámetros x, y.

7.3 test_Temporalidad

Entradas esperadas: módulo de benchmark de pytest-benchmark.

Salidas: Valores temporales de la ejecución del segmentador.

7.4 Test add user

Función que prueba la capacidad del sistema de guardar en la base de datos nuevos usuarios.

Salidas: Assert sobre si el número de usuarios en la base de datos aumentó en uno.

7.5 Test Format

Prueba que se encarga de evaluar si el formulario es capaz de manejar imágenes en distintos formatos y convertirlos a png al momento de guardar en la base de datos.

Salida: Assert binario comparando el formato de la imagen guardada con png.

7.6 Test Add Photo

Prueba encargada de evaluar el correcto funcionamiento de la base de datos. Se crea un objeto de tipo colección con una imagen asociada a este, luego se procede a hacer un save y se evalúa si ambos objetos se guardaron satisfactoriamente en la base de datos.

Salidas: Assert sobre si el número de imágenes en la base de datos aumentó en uno.

7.7 Test Size

Prueba encargada de evaluar si la base de datos respeta las dimensiones de la imagen original al guardar una instancia en el sistema. Para ello se carga una imagen de prueba llamada black.png con dimensiones de 275x183 pixeles y se compara con la original.

Salida: Assert que compara el tamaño de la imagen con el establecido.

7.8 Test Form

Prueba encargada de evaluar si el formulario para la creación de una colección de imágenes se procesa adecuadamente. Es importante destacar que por medio de este formulario es que el cliente se comunica con el servidor para subir sus imágenes, por lo tanto se prueba se los campos de entrada de datos del template validan el formulario.

Salidas: Valor booleano sobre si el formulario es válido o no.

8 Actividades de aseguramiento de calidad

Se han hecho varias actividades de aseguramiento de calidad definidas anteriormente, los cuales se pueden mencionar:

1. Evaluación de clima organizacional y problemas: en cada sprint se evalua lo que es las habilidades tecnicas, tiempo, entre otras caracteristicas de los miembros del equipo al inicio de cada sprint. Al finalizar se revisan los problemas ocurridos a lo largo del proceso de desarrollo.

2. Definición de requerimientos del sistema: Se modificaron algunos requerimientos del sistema proporcionados por el usuario en el sprint actual.

3. Construcción de pruebas unitarias: se elaboraron varias pruebas unitarias para la evaluación de las funcionalidades implementadas para el sistema.

4. Definición de métricas: se definieron algunas métricas para el análisis del sistema.