

Instituto Tecnológico de Costa Rica

Ingeniería en Computación

**IC6831-Aseguramiento de la Calidad
Avance 2: Actualización del documento SyRs
Profesor: Saúl Calderón Ramírez**

**Estudiantes:
Luis Alonzo Cascante Franco 2014159896
Olman Castillo Picado 2015148651
María Laura Pizarro Moreno 2015120626
Gabriel Venegas Castro 2013115967**

Segundo Semestre 2017

Índice

1	Introducción	3
1.1	Propósito del sistema	3
1.2	Alcance del sistema	3
1.3	Visión general del sistema	4
1.3.1	Contexto del sistema	4
1.3.2	Funciones del sistema	5
1.3.3	Características del usuario	7
2	Requerimientos del sistema	7
2.1	Requerimientos funcionales	7
2.2	Requerimientos de usabilidad	8
2.3	Requerimientos de rendimiento	8
2.4	Interfaces del sistema	8
2.4.1	Interfaces de usuario	8
2.5	Interfaces de hardware:	9
2.6	Interfaces de software:	9
2.7	Interfaces de comunicaciones:	9
2.8	Operaciones del sistema	9
2.8.1	Requisitos de integración del sistema humano	9
2.8.2	Mantenibilidad	9
2.8.3	Confiabilidad	10
2.9	Modos y estados del sistema	11
2.10	Requerimientos físicos	15
2.11	Adaptabilidad de los requerimientos	15
2.12	Condiciones ambientales	16
2.13	Seguridad del sistema	16
2.14	Gestión de la información	16
2.15	Políticas y regulaciones	16
2.16	Mantenimiento del ciclo de vida del sistema	17
2.17	Embalaje, manipulación, envío y transporte	17
2.17.1	Embalaje	17
2.17.2	Manipulación	17
2.17.3	Envío	17
2.17.4	Transporte	18
2.18	Verificación	18
3	Validación de diseño	19
4	Verificación de la codificación	21
5	Manual de administración de la configuración del software	29
5.1	Obtención de la última versión	30
5.1.1	Ingreso al repositorio	30
5.1.2	Obtención del repositorio Master	30

6	Informe del sprint	31
6.1	Métricas	31
6.1.1	Capacidad de cumplimiento de la mantenibilidad	31
6.1.2	Madurez y tolerancia a fallos	31
6.2	Unit tests	32
7	Apéndices	32
7.1	Asunciones y dependencias	32
7.2	Estándar de codificación	32
7.2.1	Estándar	32
7.2.2	Herramientas a utilizar	33
7.2.3	Referencias de las herramientas	33
7.3	Actividades del aseguramiento de la calidad	33
7.3.1	Plan de aseguramiento de la calidad del Software	33
7.3.2	Tareas de aseguramiento de la calidad	35
7.4	Diagramas de UML	37
7.4.1	Diagrama de componentes	37
7.4.2	Diagrama de clases	38

1 Introducción

1.1 Propósito del sistema

El sistema tiene como objetivo realizar el pasaporte electrónico para la dirección de Migración y extranjería, para así poder ser implementado en el aeropuerto Juan Santamaría, el paso por Peñas Blancas, Paso Canoas y Sixaola. Dicho sistema permitirá automatizar al 90% todos los trámites, para eliminar la necesidad de mostrar el pasaporte en los puntos de control. Este pasaporte, además de incluir los datos de cada sujeto (Nombre, estado civil, etc.), comprende 10 imágenes digitales de cada sujeto. Esta limitada base de datos muestral por sujeto permite la implementación de un sistema de reconocimiento automático de la identidad en los puntos de entrada principales del país. La automatización se implementará con el uso de reconocimiento facial, el cual será complementado con sistemas de reconocimiento de huellas dactilares y de iris ocular.

1.2 Alcance del sistema

Los trámites en el aeropuerto Juan Santamaría, el paso por Peñas Blancas, Paso Canoas y Sixaola, suelen ser procesos que requieren mucho tiempo, generando un incremento en la cantidad de personas que se encuentran esperando su turno, provocando que las personas tengan que llegar mucho tiempo antes de su viaje para prever el tiempo de espera.

El proyecto va a planificar, diseñar, desarrollar e implementar un sistema de reconocimiento facial llamada RecognizeMe que agilice los trámites de los lugares anteriormente mencionados. El sistema contará con 10 fotografías por persona, las cuales serán ingresadas por el encargado con los permisos respectivos. Los usuarios que no cuenten con los permisos necesarios, no podrán modificar la base de datos del sistema, sólo podrán hacer uso de la funcionalidad de reconocimiento facial.

La primera versión del sistema no contará con reconocimiento de huellas dactilares y de iris ocular, solo va a hacer reconocimiento facial y no va a contar con avisos como por ejemplo de si la persona reconocida está siendo buscada por la policía. Por el momento el sistema no podrá reconocer personas que por ejemplo estén con barba si en el sistema no se encuentra ninguna fotografía con barba, o personas que estén usando lentes mientras que en la base de datos no se encuentre una fotografía con lentes.

El objetivo general es emprender un proyecto que reemplace el uso de pasaporte con un sistema de reconocimiento facial que agilice los trámites de la dirección de Migración y extranjería.

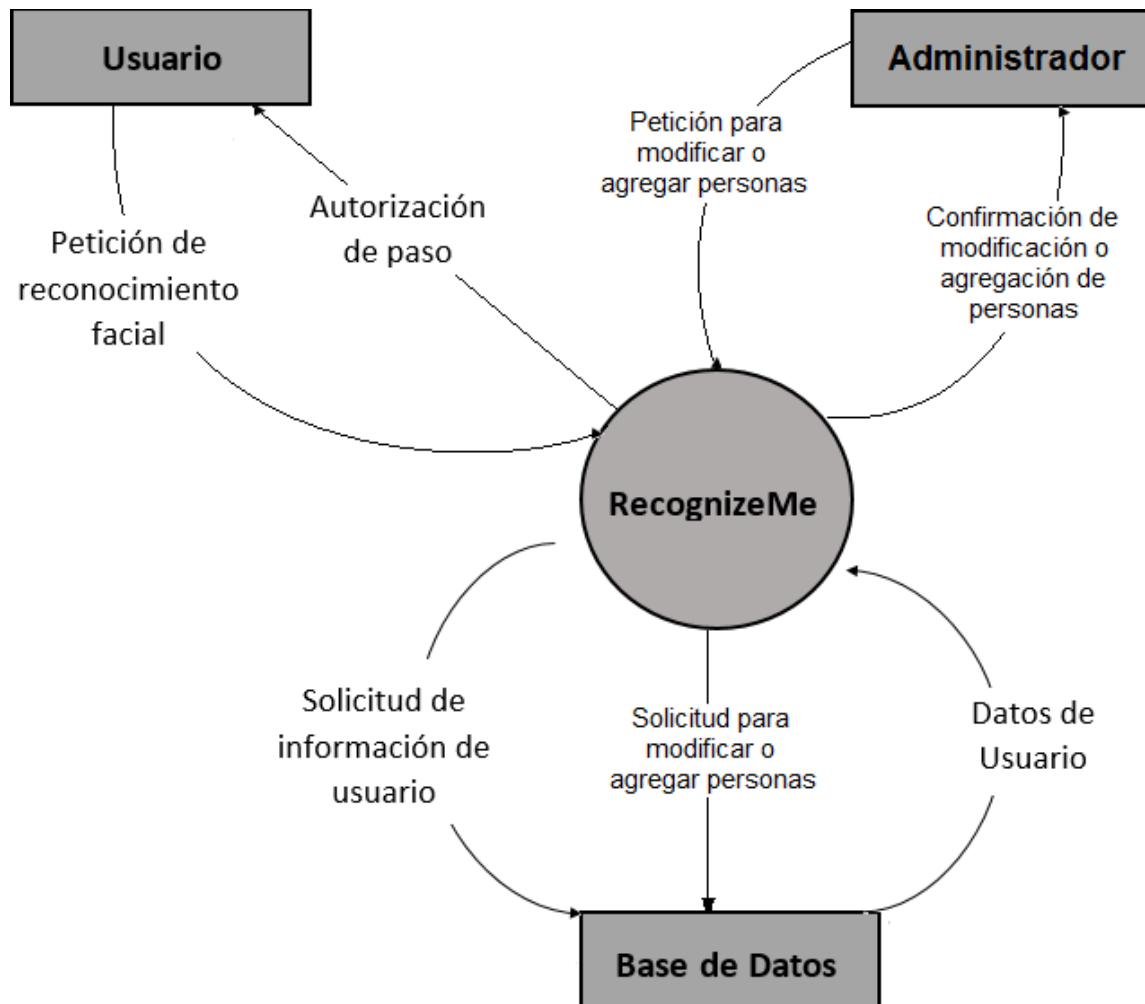
Se han identificado los objetivos que apoyan mutuamente los hitos y los resultados de este proyecto. Para lograr el éxito en el proyecto RecognizeMe, se deben cumplir los siguientes objetivos dentro del tiempo designado. El sistema tendrá el propósito de reconocer las caras de las personas que hagan uso de este, mediante algoritmos eficientes y confiables. Se contará con una base de datos que contenga el nombre y fotografías de las personas que vayan a utilizar el sistema. Se implementarán medidas de seguridad que impidan la modificación de la base de datos a personas que no tengan los permisos respectivos.

Este proyecto será aceptado una vez que el sistema sea probado exitosamente por los encargados asignados en cada centro, que vaya a hacer uso de RecognizeMe, y algunas personas externas a

los centros, que estén dispuestas a ayudar en las pruebas del sistema. Se realizarán pruebas de aseguramiento de calidad que facilitarán saber si el sistema cumple con las necesidades requeridas.

1.3 Visión general del sistema

1.3.1 Contexto del sistema



El sistema contará dos ventanas en la interfaz gráfica, en la inicial se haría el reconocimiento facial. Los usuarios harían peticiones al sistema, ya sea por medio de una cámara o ingresando una foto, y el sistema mandaría una solicitud de información de usuario a la base de datos. Luego la base de datos retornaría los datos de usuario y el sistema enviará la autorización para que el usuario pueda pasar al centro en el que se encuentre.

En la segunda ventana se haría el entrenamiento del sistema. En esta sección los administradores ingresarán a nuevas personas o harán modificaciones de personas que se encuentren en el sistema.

La base de datos recibiría la información y la almacenaría, luego le indicaría al sistema que las imágenes se ingresaron exitosamente o si hubo algún error.

1.3.2 Funciones del sistema

Descripción y prioridad

- Petición de ingreso al centro: Todo usuario sin importar los permisos va a poder hacer uso de esta funcionalidad. Hay dos formas en la que se puede hacer, ingresando una imagen que cumpla con las características necesarias y usando una cámara que toma en tiempo real la fotografía del usuario. Esta funcionalidad es de prioridad media.
- Entrenamiento del sistema: Sólo los usuarios con permisos podrán acceder a la ventana de entrenamiento. En la interfaz de entrenamiento se ingresarán nuevas personas al sistema o modificarán datos ya existentes. Esta funcionalidad es de prioridad media.
- Cargar imágenes almacenadas en una dirección provista por el usuario para el entrenamiento: El conjunto de imágenes provisto debe contener una carpeta por cada sujeto, y se debe almacenar el nombre de tal carpeta como la etiqueta de tal sujeto o clase. Esta funcionalidad es de prioridad alta.
- Entrenar el sistema generando auto-caras y la proyección de las muestras en el nuevo espacio formado por tales auto-caras, usando el algoritmo supervisado del centroide más cercano. Tanto las auto-caras como las proyecciones se almacenarán usando algún esquema de almacenamiento. Esta funcionalidad es de prioridad alta.
- Cargar un conjunto de muestras de prueba, para medir la precisión del sistema. Esta funcionalidad es de prioridad alta.

Secuencia de respuestas:

- Carga de imágenes para el entrenamiento por usuario:
 - Usuario: Indica la dirección de la carpeta con las imágenes de entrenamiento.
 - Sistema: Recibe las imágenes de la dirección ingresada.
 - * Si cumple con las características: Guarda las imágenes en la base de datos.
 - * Si no cumple con las características: No guarda las imágenes y muestra mensaje de error
- Entrenar el sistema generando auto-caras:
 - Usuario: Ingresa imagen.
 - Sistema: Recibe la imagen.
 - * Si cumple con las características: Genera auto-caras.
 - * Si no cumple con las características: No genera auto-caras y muestra mensaje de error.
- Cargar muestras de prueba:

- Usuario: Inicia el sistema.
- Sistema: Empieza a cargar las muestras de prueba.
 - * Si cumple con las características: Carga las muestras exitosamente.
 - * Si no cumple con las características: No carga las muestras.
- Petición de ingreso al centro
 - Usuario: Ingresa imagen al sistema
 - Sistema: Verifica que la imagen cumpla con las características necesarias.
 - * Si cumple con las características: Retorna la información del usuario y permite el acceso.
 - * Si no cumple con las características: No permite el acceso y muestra mensaje de error.
- Ingresar una nueva persona al sistema:
 - Usuario: Ingresa nombre e imágenes al sistema
 - Sistema: Verifica que los datos ingresados sean correctos.
 - * Si son correctos: Se agrega una nueva persona al sistema.
 - * Si son incorrectos: Se notifica al usuario que hay un error.
- Modificar usuario existente:
 - Usuario: Modifica los campos que desea modificar.
 - Sistema: Verifica que los campos se llenen correctamente.
 - * Si son correctos: Se actualiza la información de la persona modificada.
 - * Si son incorrectos: Se notifica al usuario que hay un error.

Requerimientos Funcionales:

- Carga de imágenes para el entrenamiento por usuario:
 - Se debe de llenar el campo de:
 - * Dirección Si no se llenan correctamente: Se notifica al usuario que hay un error.
- Entrenar el sistema generando auto-caras:
 - Se debe de llenar el campo de:
 - * Imagen (ya sea de forma manual o con el uso de la cámara) Si no se llenan correctamente: Se notifica al usuario que hay un error.
- Cargar muestras de prueba:
 - Solo se debe de iniciar el sistema.
- Petición de ingreso al centro

- Se debe de llenar el campo de:
 - * Imagen (ya sea de forma manual o con el uso de la cámara) Si no se llenan correctamente: Se notifica al usuario que hay un error.
- Ingresar una nueva persona al sistema:
 - Se debe llenar los campos de:
 - * Nombre
 - * Imagen (La imagen debe cumplir con las características necesarias) Si no se llenan correctamente: Se notificará al usuario que hay un error.
- Modificar usuario existente:
 - Se debe llenar los campos de:
 - * Nombre
 - * Imagen (La imagen debe cumplir con las características necesarias) Si no se llenan correctamente: Se notificará al usuario que hay un error

1.3.3 Características del usuario

El sistema contará con 2 tipo de usuario.

- Usuario: van a ahacer uso de la funcionalidad de reconocimiento facial.
- Administrador: Los usuarios con permisos son los que van a poder entrenar el sistema y agregar nuevos usuarios.

2 Requerimientos del sistema

2.1 Requerimientos funcionales

El sistema cuenta con los siguientes requerimientos funcionales:

- Ingresar imagen
 - Se debe de ingresar una imagen de formato y tamaño válidos
 - Si no se ingresa una imagen con las características necesarias el sistema va a mostrar un mensaje de error
 - Si no se ingresa ninguna imagen el sistema no va a proceder
- Tomar una foto
 - Se debe colocar la cara de la forma correcta para capturar la imagen
 - Si no se toma correctamente la foto, el sistema no va a reconocer a la persona.

2.2 Requerimientos de usabilidad

- La disponibilidad de RecognizeMe estará sujeta a las decisiones pertinentes hechas por el cliente
- EL software será adaptable a cualquier ordenador que cuente con Django 1.11 y Python 3.4.4.
- RecognizeMe será flexible en cuanto a la localización de sus usuarios, pues estos podrán estar ubicados en cualquier lugar de Costa Rica.
- La usabilidad de RecognizeMe tendrá un nivel alto, pues tanto su interfaz gráfica como sus funcionalidades serán fáciles de aprender a utilizar, y fáciles y eficientes al usarlas con experiencia.
- RecognizeMe será un sistema eficiente, que llevará a cabo peticiones del usuario sin agregar información ni pasos innecesarios.

2.3 Requerimientos de rendimiento

El sistema RecognizeMe, aunque no se especifique un tiempo de reacción o desempeño específico, debe actuar relativamente rápido, de manera que se garantice que el usuario se sienta cómodo y satisfecho con la actuación óptima de la aplicación. El tiempo de reacción solo debe verse afectado por la velocidad del dispositivo en cuestión, o la calidad de datos que se encuentren en la base de datos.

- Al momento de realizar un proceso, este no debe de sobrepasar el 50% del uso del CPU, la carga en memoria no debe superar a los 20 Kb, 30 Kb.
- El 95% de las transacciones deben de realizarse en menos de un minuto.
- Se espera que el tiempo mínimo de vida sea de 5 años. Manteniendo el mismo rendimiento inicial.
- El sistema estará capacitado para funcionar 24 horas por día sin ningún problema de rendimiento.
- No es necesario tener experiencia para aprovechar al máximo las capacidades del sistema. Por lo tanto no va a afectar al rendimiento quien esté operando el sistema.
- El sistema es muy sencillo por lo que no hay requerimientos para su uso, por lo tanto el rendimiento siempre va a ser el mismo.

2.4 Interfaces del sistema

2.4.1 Interfaces de usuario

Ingreso de imagen: Cualquier usuario podrá ingresar una imagen o tomarse una foto, para realizar el reconocimiento facial.

Modificación de la base de datos: Los administradores podrán ingresar nuevos usuarios en cualquier momento.

2.5 Interfaces de hardware:

El programa de RecognizeMe funciona en cualquier ordenador que cuente con Django 1.11 y Python 3.4.4.

2.6 Interfaces de software:

El sistema RecognizeMe maneja los datos desde una carpeta en el ordenador para poder obtener los datos necesarios para hacer uso de la aplicación.

2.7 Interfaces de comunicaciones:

El sistema de RecognizeMe no tendrá conexiones a internet en esta primera versión. Se planea hacer las conexiones para las siguientes versiones.

2.8 Operaciones del sistema

2.8.1 Requisitos de integración del sistema humano

El sistema debe de contar con una interfaz, que sea eficiente de utilizar y que se adecue a diferentes necesidades, ya que en el entorno en el que se utilizará, es un entorno en el cual será usada por diferentes tipos de usuarios con distintas necesidades. Entre las consideraciones se ha tomado en cuenta:

- El crear una aplicación con una letra de grande para permitirle al usuario, con poca capacidad de visión el leer las indicaciones.
- Al ser utilizada por gran cantidad de usuarios se planea el elaborar una aplicación sencilla y agradable para la vista y no muy llamativa para evitar distracciones en el usuario y agilizar el proceso.

2.8.2 Mantenibilidad

El sistema será utilizado en un contexto en el cual el tiempo de inactividad del sistema debe ser mínimo y en caso de que se presentará el tiempo de respuesta debe ser eficiente y de la manera más pronta. Se establecen los siguientes requisitos de mantenimiento, para establecer un contexto de respuesta ante la necesidad de darle mantenimiento al sistema.

- Tiempo: en brindar mantenimiento al sistema considerando la importancia de este de estable que los tiempos de mantenimiento serán de la duración mínima posible.

	Mantenimiento por modificación	Mantenimiento preventivo	Mantenimiento por fallos
Tiempo de respuesta	0	0	1 hora
Tiempo promedio de inactividad	2 horas	1 hora	3 horas
Tiempo máximo de inactividad	4 horas	3 horas	6 horas

- Tarifa:

	Mantenimiento por modificación	Mantenimiento preventivo	Mantenimiento por fallos
Costo por persona	12000 colones por hora	12000 colones por hora	14000 colones por hora
Frecuencia	Cada vez que sea solicitado	Cada 4 meses	Cada vez que se presente un fallo

- Complejidad:

	Mantenimiento por modificación	Mantenimiento preventivo	Mantenimiento por fallos
Número de personas	4	4	4

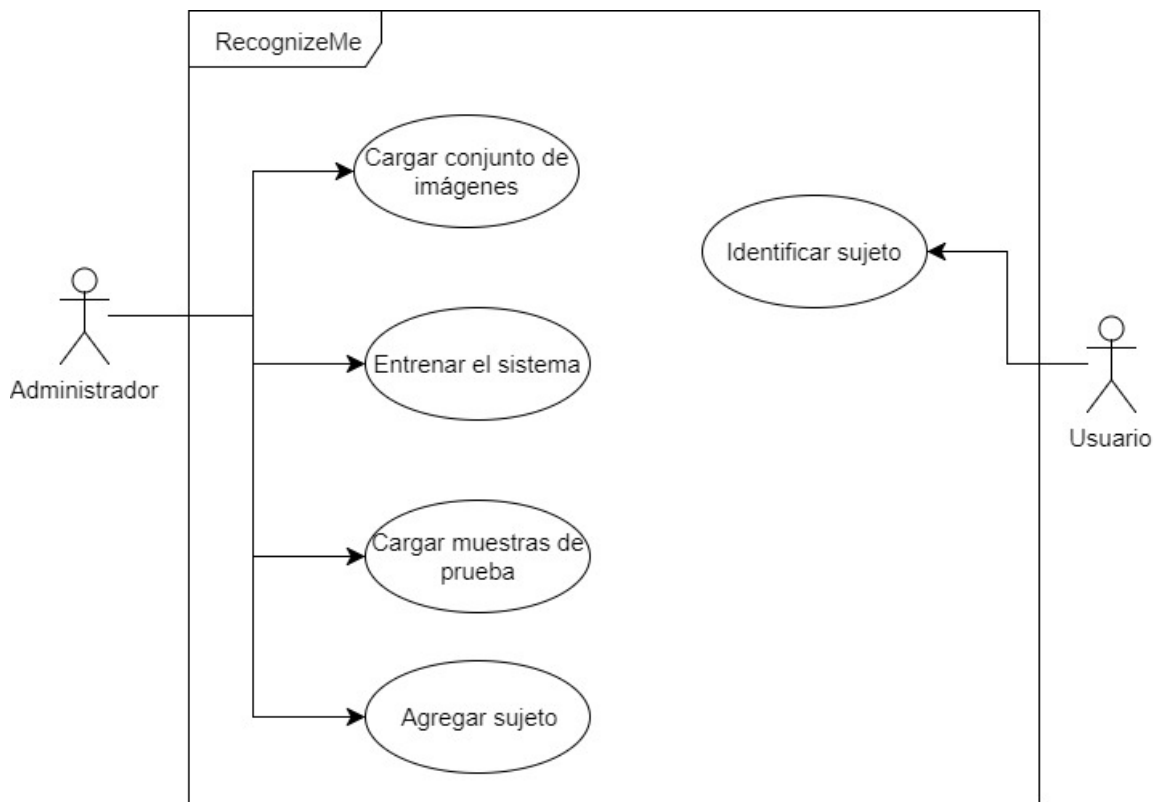
- Índices de acción:

	Mantenimiento por modificación	Mantenimiento preventivo	Mantenimiento por fallos
Costo por persona	24000 colones	12000 colones	42000 colones
Costo total	96000 colones	48000 colones	168000 colones

2.8.3 Confiabilidad

Se establece un valor de fiabilidad de 98% para el periodo de un año, en la funcionalidad de reconocimientos de sujetos que será ejecutada dentro de los aeropuertos internacionales y los puntos de entrada por vía terrestre.

2.9 Modos y estados del sistema

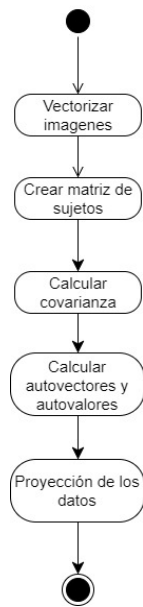


Estados

- Cargar conjunto de imagenes



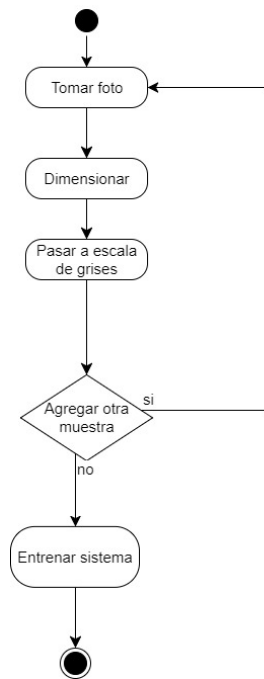
- Entrenar sistema



- Cargar muestras de prueba



- Agregar sujeto



- Identificar sujeto



2.10 Requerimientos físicos

El sistema RecognizeMe estará ubicado en un servidor central desde el cual se brinda el servicio a los lugares que lo requieran. Considerando el volumen de datos y la necesidad de que sea un sistema de respuesta rápida, se pretende utilizar las herramientas y servidores, que aseguren el funcionamiento correcto del sistema. Las funcionalidades serán administradas desde dos tipos de plataformas:

- La funcionalidad de identificar sujeto, será accesada desde una computadora con pantalla táctil con un navegador.
- Las funcionalidades del administrador serán utilizadas desde un ordenador ubicado en cada uno de los puntos de ingreso al país.

2.11 Adaptabilidad de los requerimientos

Al tener almacenada una gran cantidad de información en el sistema, es necesario que se tenga la posibilidad de aumentar el tamaño del espacio de procesamiento, además por lo importante que son los datos en el sistema se debe contar con una manera de generación de respaldos de la información la cual se debería mantener actualizado en no más de un periodo de 24 horas.

2.12 Condiciones ambientales

El sistema no cuenta con alguna condición ambiental específica que dificulte la ejecución correcta de la aplicación.

2.13 Seguridad del sistema

La seguridad del sistema será gestionada diferente según sea el tipo de permiso que tenga la persona que desea ingresar al sistema:

- Permisos de usuario: tendrá únicamente acceso a la funcionalidad de identificar usuario, no presenta ninguna restricción.
- Permisos de administrador: se encargará de las funcionalidades de entrenamiento, y cargar imágenes contará con un método de ingreso de usuario y contraseña diferentes para cada uno de los puntos de acceso de país.

2.14 Gestión de la información

El sistema se comunicará diariamente con una base de datos en internet para ingresar la nueva información registrada en el día y recibir la nueva información ingresada en los otros centros que cuenten con el sistema RecognizeMe.

El sistema aceptará imágenes de cualquier tamaño y color. Si en la imagen la cara del sujeto no está centrada correctamente el sistema no podrá reconocer a la persona. En el momento de reconocer la cara del usuario el sistema va a retornar el nombre del sujeto y luego permitirá el acceso al centro. Para poder realizar este proceso el sistema contará con una base de datos que contiene 10 imágenes por persona registrada.

Cuando un administrador desee registrar un nuevo sujeto debe de ingresar 10 imágenes y el nombre del mismo. La base de datos almacenará la información en una carpeta con el nombre del sujeto. Si se desea modificar algún sujeto se debe seleccionar la imagen o imágenes a reemplazar o ingresa la actualización del nombre del sujeto.

La información que maneje el sistema no pasara por algún protocolo de seguridad ya que la información se maneja localmente. Cuando se haga el respaldo diario la información se encriptará por medida de seguridad.

2.15 Políticas y regulaciones

1. El sistema estará solo en español, se planea implementar otros idiomas en futuras versiones.
2. La información del sistema solo será manipulada por los administradores.
3. El sistema es seguro de usar por lo que no hay medidas de salud ni seguridad.

4. RecognizeMe debe de estar supervisado en todo momento por si el usuario necesita ayuda o por si sucede alguna anomalía.
5. El hardware que necesita el sistema es muy básico por lo que no van a haber problemas de toxinas o radiación.
6. Cada 24 horas se hará un respaldo de la base de datos.
7. En esta primera versión el sistema solo maneja reconocimiento facial.El reconocimiento ocular y de huellas se implementará en futuras versiones.

2.16 Mantenimiento del ciclo de vida del sistema

El mantenimiento del ciclo de vida del sistema se llevará a cabo por medio de la puesta en práctica de:

1. Revisiones de diseño y de requerimientos.
2. Seguimiento del avance y desempeño de los productos de software.
3. Seguimiento del avance y desempeño de los procesos de software.
4. Corroboración del cumplimiento de los estándares establecidos.
5. Actividades establecidas en la sección 3.3 del presente documento durante y después de la ejecución de los sprints correspondientes.

2.17 Embalaje, manipulación, envío y transporte

2.17.1 Embalaje

El sistema se guardará en un CD y en una memoria USB para luego ser instalado en el centro que sea solicitado.

2.17.2 Manipulación

El programa será manipulado por personas autorizadas de la empresa. Ya sea el embalaje, la manipulación, el envío y el transporte serán regulados por personas autorizadas.

2.17.3 Envío

El programa no será enviado por correo electrónico ni físico. Una persona autorizada se encargará de llevar personalmente el programa y de instalar en todos los ordenadores estipulados previamente en el contrato.

2.17.4 Transporte

La persona encargada de hacer el envío del programa se transportará en el vehículo de su elección mientras que no atrase la entrega del producto.

2.18 Verificación

Se debe verificar que cada uno de los trozos de código que resuelven los requerimientos funcionales tengan un correcto funcionamiento, conforme con la calidad del producto. Esto se hará por medio de pruebas unitarias o 'unittest'.

3 Validación de diseño

Validación de Diseño	
Unidad de Diseño	SyRS
Vistas: <ul style="list-style-type: none"> • administrador Controladores: <ul style="list-style-type: none"> • Ui.administrador • Facade • Facade.administrador • Control • gestorMuestra • DAOBDmuestral Clases: <ul style="list-style-type: none"> • muestra • sujeto • imagen 	Carga de imágenes para el entrenamiento por usuario.
Vistas: <ul style="list-style-type: none"> • administrador Controladores: <ul style="list-style-type: none"> • Ui.administrador • Facade • Facade.administrador • Control • gestorMuestra • gestorPCA • DAOBDpca Clases: <ul style="list-style-type: none"> • PCA • muestra • sujeto • imagen • Proyeccion • Clasificación (centrodide y k-vecinos) 	Entrenar el sistema generando auto-caras.

Validación de Diseño	
Unidad de Diseño	SyRS
Vistas: <ul style="list-style-type: none"> • administrador Controladores: <ul style="list-style-type: none"> • Ui_administrador • Facade • Facade_administrador • Control • gestorMuestra • DAOBDmuestreal • gestorPCA • DAOBDpca Clases: <ul style="list-style-type: none"> • muestra • sujeto 	Cargar muestras de prueba.
Vistas: <ul style="list-style-type: none"> • usuario Controladores: <ul style="list-style-type: none"> • Ui_usuario • Facade • Facade_usuario • Control • gestorPCA • DAOBDpca Clases: <ul style="list-style-type: none"> • PCA • proyección • clasificación • muestra • sujeto 	Petición de ingreso al centro.

Validación de Diseño	
Unidad de Diseño	SyRS
Vistas: <ul style="list-style-type: none"> • administrador Controladores: <ul style="list-style-type: none"> • Ui_administrador • Facade • Facade_administrador • Control • gestorMuestra • DAOBDmuestreal Clases: <ul style="list-style-type: none"> • sujeto • muestra • imagen 	Ingresar una nueva persona al sistema.
Vistas: <ul style="list-style-type: none"> • administrador Controladores: <ul style="list-style-type: none"> • Ui_administrador • Facade • Facade_administrador • Control Clases: <ul style="list-style-type: none"> • sujeto • imagen 	Modificar usuario existente.

La tabla anterior muestra la relación que tienen las unidades de diseño del actual proyecto con los requerimientos planteados. Cabe mencionar que cada unidad de diseño ejecuta una cantidad limitada de funciones, y que se requiere el conjunto de estas para cumplir con los requerimientos estipulados en el presente documento SyRS.

4 Verificación de la codificación

Se escogió la herramienta de SonarQube para verificar el estándar de codificación en conjunto con la herramienta Jenkins en el sistema operativo Windows. SonarQube es una plataforma de código abierto para realizar revisiones automáticas con análisis estáticos de código para detectar errores, olores de código y vulnerabilidades de seguridad en lenguajes de programación de más de 20. Para poder hacer uso se deben de seguir los siguientes pasos para su instalación y configuración.

Instalación de Jenkins:

1. Instalar Jenkins de <https://jenkins.io/download/>

Long-term Support (LTS)

LTS (Long-Term Support) releases are chosen every 12 weeks from the stream of regular releases as the stable release for that time period. [Learn more...](#)

[Changelog](#) | [Upgrade Guide](#) | [Past Releases](#)

Deploy Jenkins 2.73.1



Download Jenkins 2.73.1 for:

Docker
FreeBSD
Gentoo
Mac OS X
OpenBSD
openSUSE
Red Hat/Fedora/CentOS
Ubuntu/Debian
Windows
Generic Java package (.war)

2. Extraiga el zip y ejecute el.msi
3. Diríjase a <http://localhost:8080/>
4. Siga los pasos que Jenkins va mostrando en pantalla.

Instalación de SonarQube:

1. Extraiga el zip en "C: \sonarqube" o " / etc / sonarqube", la distribución de SonarQube una vez descargado. <https://www.sonarqube.org/downloads/>
2. Inicie el servidor SonarQube: C:\sonarqube\bin\windows-x86-xx\StartSonar.bat
3. Inicie sesión en <http://localhost:9000> con las credenciales del administrador del sistema (admin / admin) y siga el tutorial para analizar su primer proyecto.

Configuración de SonarQube:

1. Luego de iniciar sesión se visualizará el siguiente cuadro en donde debe de ingrear un nombre para generar el token.

2. Luego de que se genera el token indique el lenguaje del proyecto.
3. Indique el sistema operativo en donde va a correr su código.
4. Por último se debe definir una llave única para el proyecto.
5. Luego de ingresar toda la información anterior se debe descargar el SonarQube Scanner y se debe agregar la dirección del folder 'bin' a 'path' en las variables de entorno del sistema.
6. Se debe de abrir el cmd y colocarse en la dirección en donde se encuentre el archivo 'sonar-scanner.bat' para luego poder ejecutarlo.
 - Antes de ejecutar el archivo se debe de actualizar el archivo 'sonar-scanner.properties' con los datos del cuadro negro.

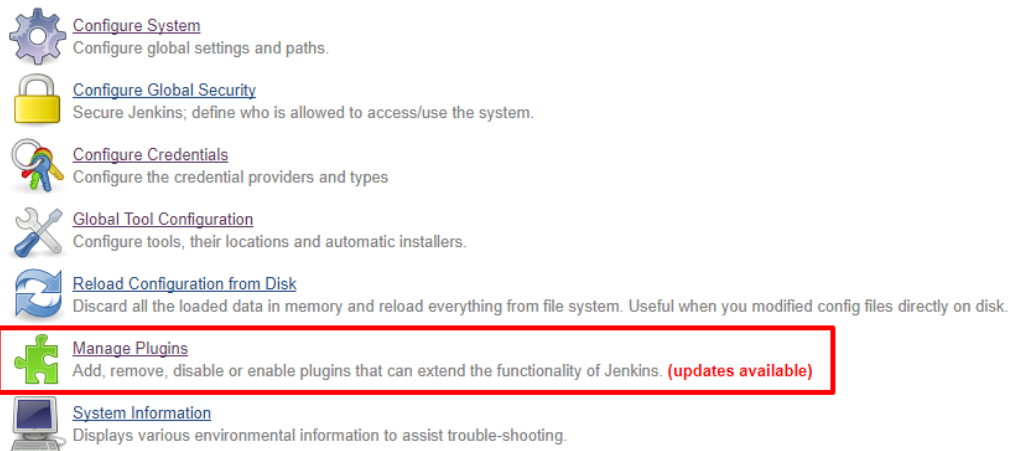
Configuración de Jenkins con git y SonarQube:

1. Estando en la página principal de Jenkins diríjase a 'Manage Jenkins'.




2. Diríjase a la opción 'Manage Plugins'.


Manage Jenkins





3. Luego en la pestaña de 'Available' buscar el plugin llamado 'SonarQube Scanner for Jenkins' y 'Git plugin' cuando se esté instalando seleccione el cuadro que dice que se reinicie Jenkins después de instalar.
4. Luego de que se reinicie se va a volver a mostrar la ventana principal, volvemos a dar click en 'Manage Jenkins'.
5. Le damos click a la opción 'Configure System'.


Manage Jenkins


[Configure System](#)
Configure global settings and paths.


[Configure Global Security](#)
Secure Jenkins; define who is allowed to access/use the system.

[Configure Credentials](#)
Configure the credential providers and types

[Global Tool Configuration](#)
Configure tools, their locations and automatic installers.

[Reload Configuration from Disk](#)
Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.

[Manage Plugins](#)
Add, remove, disable or enable plugins that can extend the functionality of Jenkins. **(updates available)**

[System Information](#)
Displays various environmental information to assist trouble-shooting.

6. En la sección de ‘SonarQube servers’ se deben llenar los siguientes campos siguiendo las sugerencias que se encuentran debajo de cada cuadro. El ‘authentication token’ es el token generado cuando se configuró el SonarQube. Luego de llenar todos los campos se salvan los cambios.

SonarQube servers

Environment variables

☒ Enable injection of SonarQube server configuration as build environment variables
If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

SonarQube installations

Name

sonarqube

Server URL

http://localhost:9000/sonar

Default is http://localhost:9000

Server version

5.3 or higher

Configuration fields depend on the SonarQube server version.

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

SonarQube account login

SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.

SonarQube account password

SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.

Advanced...








Delete SonarQube

Save

Apply



7. De nuevo se va a mostrar la pantalla inicial y se deberá ingresar a ‘Manage Jenkins’ otra vez. Esta vez hay a ingresar a ‘Global Tool Configuration’.

Manage Jenkins

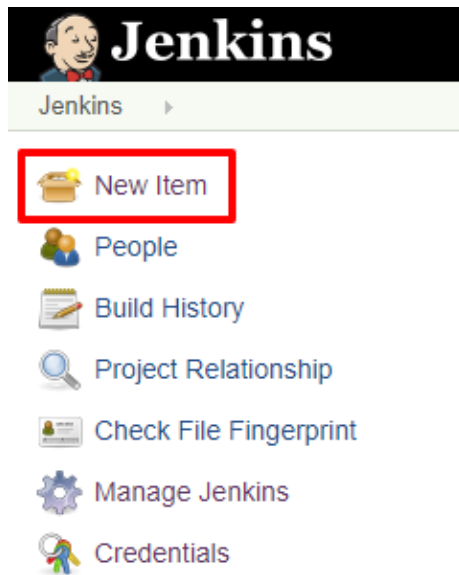
-  [Configure System](#)
Configure global settings and paths.
-  [Configure Global Security](#)
Secure Jenkins; define who is allowed to access/use the system.
-  [Configure Credentials](#)
Configure the credential providers and types
-  [Global Tool Configuration](#)
Configure tools, their locations and automatic installers.
-  [Reload Configuration from Disk](#)
Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
-  [Manage Plugins](#)
Add, remove, disable or enable plugins that can extend the functionality of Jenkins. (updates available)
-  [System Information](#)
Displays various environmental information to assist trouble-shooting.

8. En la sección de 'SonarQube Scanner' se deben llenar los siguientes campos. El 'SONAR_RUNNER_HOME' es la dirección en donde se encuentran los archivos del scanner. Al finalizar se deben guardar los cambios.

SonarQube Scanner

SonarQube Scanner installations	
	<div>SonarQube Scanner</div> <div>Name <input type="text" value="sonar Scanner"/></div> <div>SONAR_RUNNER_HOME <input type="text" value="C:\Program Files (x86)\sonar-scanner-3.0.3.778-windows"/></div> <div><input type="checkbox"/> Install automatically</div> <div></div>
<div>Delete SonarQube Scanner</div>	
<div>Add SonarQube Scanner</div>	

9. Luego se debe proceder a crear un nuevo ítem.



10. Se le asigna un nombre al ítem y se selecciona la opción de 'Freestyle project'.
11. En la pestaña 'Source Code Management' se selecciona git y se completan los campos correspondientes.

The image shows the 'Source Code Management' configuration form in Jenkins. The form has a sidebar on the left with radio buttons for 'None' and 'Git' (selected). The main area is divided into three sections: 'Repositories', 'Branches to build', and 'Repository browser'. The 'Repositories' section contains a 'Repository URL' field with the value 'https://github.com/Hamdog28/Asdecalidad-1.git', a 'Credentials' dropdown menu with 'Hamdog/*****' selected, and an 'Add' button. There are also 'Advanced...' and 'Add Repository' buttons. The 'Branches to build' section contains a 'Branch Specifier (blank for 'any')' field with the value '*/master' and an 'Add Branch' button. The 'Repository browser' section contains a dropdown menu with '(Auto)' selected. At the bottom of the form are 'Save' and 'Apply' buttons.

12. En la pestaña 'Build' se agrega una configuración seleccionando la opción de SonarQube Scanner.

13. Se llenan los espacios en blanco. Se debe brindar la dirección del archivo 'sonar-scanner.properties' y la información de ese mismo archivo en el cuadro continuo. Al final se deben guardar los cambios realizados.

Build

Execute SonarQube Scanner

Task to run

JDK

JDK to be used for this SonarQube analysis

Path to project properties

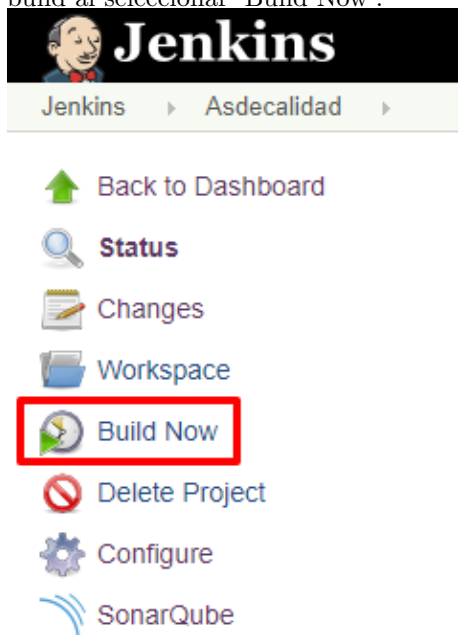
Analysis properties

```
sonar.projectKey=asdecalidad
sonar.projectName=asdecalidad
sonar.sources=
sonar.host.url=http://localhost:9000
sonar.login=c47a9c01cdeebd2085d22fad42ce8d3656d94a1b
sonar.projectVersion=1.0
sonar.sourceEncoding=UTF-8
```

Additional arguments

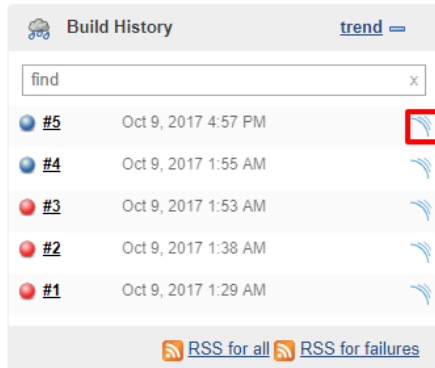
JVM Options

14. Luego se puede ver que nuestro item ha sido creado y ahora se puede proceder a realizar un build al seleccionar 'Build Now'.

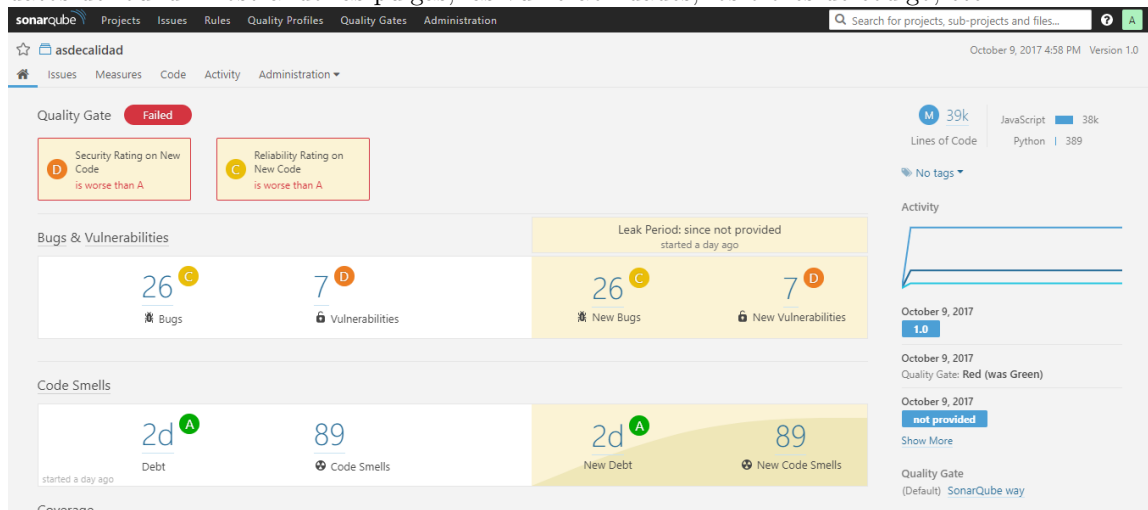


15. Una vez que finaliza exitosamente el build se pueden ver los datos al hacer click en las líneas

del lado derecho.



16. Por último se va a redireccionar la página a la de SonarQube en donde se van a desplegar los datos del build. Mostrando las pulgas, las vulnerabilidades, los olores de código, etc.



5 Manual de administración de la configuración del software

El presente manual pretende mostrarle al usuario cómo obtener la última versión estable del sistema de software.

La totalidad del sistema está presente en un repositorio de Github, en donde se puede llevar un control sobre las funcionalidades que se le van añadiendo al sistema con cada avance o “sprint”.

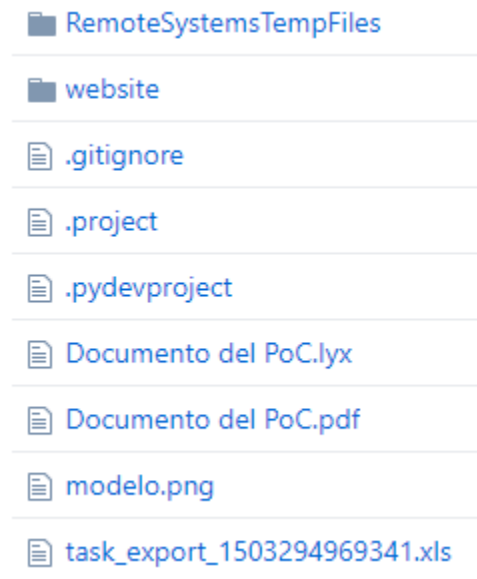
Mediante la herramienta Github, se puede trabajar en distintas partes del sistema de software a la vez de manera separada, y cuando alguna unidad está completada, se une con la instancia principal del sistema, la cual posee todos los cambios estables hasta la fecha.

A continuación se muestra cómo obtener esa última versión de la que estamos hablando.

5.1 Obtención de la última versión

5.1.1 Ingreso al repositorio

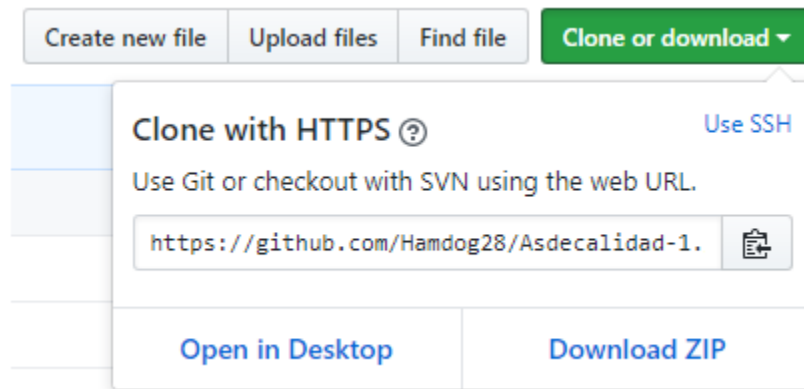
Para ingresar al repositorio solamente hace falta entrar a la página <https://github.com/Hamdog28/Asdecalidad-1>, en donde se encuentra almacenado el sistema de software. Una vez en dicha página web, podrá ver ciertos archivos, similar a como se muestra a continuación:



El proyecto en desarrollo se encuentra en la carpeta `website`. Nota: Necesita tener acceso al repositorio, si no lo posee, puede pedirlo a algún miembro del equipo de trabajo.

5.1.2 Obtención del repositorio Master

Como se mencionó anteriormente, el repositorio almacena la última versión en la rama “Master”, misma que puede ser descargada mediante el botón de “Clone or download” presente en la misma página.



6 Informe del sprint

6.1 Métricas

6.1.1 Capacidad de cumplimiento de la mantenibilidad

Gracias a la herramienta pep8 (tiene el mismo nombre del estándar que verifica), podemos obtener un conteo de los warnings y errores detectados por la herramienta para cada uno de los archivos. En este caso no nos interesan tanto los errores, pues serían mucho más evidentes, sino los warnings ya que son warnings de estilo de apego al estándar pep8, que por convención fue el que nos comprometimos a usar para estandarizar nuestro código y al hacerlo facilitar la mantenibilidad del mismo. Los resultados obtenidos para la última versión del software durante el sprint fue un promedio de 4 warnings por archivo con 0 errores por archivo.

Para utilizar la herramienta primero debe:

- Instalarla mediante pip con el siguiente comando: `pip install pep8`
- Posterior a esto debe obtener la dirección absoluta del archivo a evaluar y llamar a pep8 con:
`pep8 -count ¡direccion!`

Esto le retornara el conteo total de errores y warnings.

6.1.2 Madurez y tolerancia a fallos

Por el estándar utilizado y además por buenas prácticas del equipo de trabajo, se contabilizó una tasa de funciones con manejo de fallos del 100%. No es necesario pero como nota si alguna función no lo tuviera y lo necesitara la herramienta pep8 nos tiraría un pequeño warning.

6.2 Unit tests

7 Apéndices

7.1 Asunciones y dependencias

1. Se supone que los ordenadores en donde se ejecutará el sistema contarán con Django 1.11 y Python 3.4.4. Cualquier otra versión podría generar errores.
2. Se supone que el usuario ya tiene los conocimientos básicos para poder usar un ordenador.
3. Se supone que se hará uso de archivos desde el ordenador para desarrollar el sistema de base de datos correspondiente a la aplicación.
4. Se supone que el equipo de trabajo continuará siendo compuesto por 4 desarrolladores que programarán y pondrán a prueba el software, y crearán la documentación adjunta, en conjunto con el cliente, que brindará información en caso de que existan consultas sobre el proyecto.
5. Se supone que el usuario va a saber español.
6. Se supone que el usuario no va a tener ninguna discapacidad física ni mental que le impidan hacer uso de la aplicación. Si el usuario contara con alguna condición que lo impida hacer uso del sistema, va a haber una persona que se encargue de ayudar al usuario a utilizar el sistema.

7.2 Estándar de codificación

7.2.1 Estándar

Para tomar la decisión de cuál estándar utilizar en este proyecto se tomaron en cuenta los siguientes estándares: PEP8, CKAN 2.7 y Django Coding Style. Después de evaluar cada uno, decidimos utilizar el Django Coding Style por las siguientes razones:

- CKAN 2.7 fue descartado pues es un estándar para Python 2.7 y aunque la mayoría de las funcionalidades son iguales al 3.4, se descarta escogiendo compatibilidad.
- PEP8 no fue descartado del todo, pues está incluido en los otros 2 estándares, es solo que parece ser el estándar *básico* de Python.
- Django Coding Style: este es un estándar extraído directamente de la documentación de Django. Para nosotros es el mejor porque: es un estándar especializado en Django la tecnología que utilizamos, contiene al estándar PEP8 excepto donde el estándar Django diga explícitamente lo contrario, y porque no es un estándar especializado en una versión de Python específica. Además de que nos brinda un estándar para imports tanto de python como Django y también para la utilización de sus librerías.

7.2.2 Herramientas a utilizar

Existen varias herramientas para revisar la conformidad con el estándar PEP8 en general y además para la calidad del código en general. Utilizaremos las siguientes:

- pep8: comprueba el código Python contra algunas de las convenciones de estilo en PEP 8. Solo se deben realizar limpiezas de estilo en la branch master, para evitar *merge conflicts*.
- pylint: analiza código Python en busca de bugs o señales de calidad pobre de código.
- pyflakes: analiza el código Python en busca de errores.
- flake8: combina pep8 y pyflakes en una sola herramienta.

7.2.3 Referencias de las herramientas

References

- [1] PEP 8 – Style Guide for Python Code,
<https://www.python.org/dev/peps/pep-0008/>
- [2] Django Coding style,
<https://docs.djangoproject.com/en/dev/internals/contributing/writing-code/coding-style/>

7.3 Actividades del aseguramiento de la calidad

7.3.1 Plan de aseguramiento de la calidad del Software

Propósito El propósito de este plan de aseguramiento de la calidad de software es plasmar en un documento la organización de los miembros del equipo de desarrollo de software y los roles y responsabilidades asignados a cada uno para el aseguramiento de la calidad del software.

Administración Los roles de los encargados del aseguramiento son:

- Administrador de Proyectos
- Ingeniero de Software
- Desarrollador
- Verificador
- Analista

Organización Los distintos roles presentes en el proyecto se organizan de la siguiente manera:



Roles y responsabilidades A continuación se detallan las responsabilidades asociadas a cada rol:

Rol	Responsabilidad
Administrador de Proyectos	Encargado de la gestión de la calidad del proyecto y de la comunicación sobre errores en el plan de calidad.
Ingeniero de Software	Encargado del diseño de la arquitectura del proyecto, conforme con los estándares de calidad establecidos para el proyecto
Desarrollador	Encargado de la codificación e implementación de estándares de calidad dentro del código del proyecto, así como del uso de herramientas establecidas para el aseguramiento de la calidad del mismo.
Analista	Encargado de la recolección y análisis de los requerimientos planteados por el cliente para el proyecto actual y organiza los datos en función de los estándares de calidad establecidos para el proyecto.
Verificador	Encargado de las revisiones de software en cada una de las etapas del proyecto.

Metodología La metodología a utilizar para el aseguramiento de la calidad es la planteada por el estándar IEEE 730-2002 para Planes de Aseguramiento de Calidad de Software.

7.3.2 Tareas de aseguramiento de la calidad

Actividades del aseguramiento Definición de actividades de aseguramiento de la calidad para asegurar la calidad del software:

Actividades	Detalles	Encargados
Evaluación de cumplimiento del diseño respecto a los requerimientos	Una vez finalizado el diseño se procede a evaluar la conformidad del mismo con respecto a los requerimientos recopilados para el sistema de software. Esta actividad aplica hasta que el diseño sea definitivo y cumpla con los requerimientos.	<ul style="list-style-type: none"> • Administrador de proyectos • Ingeniero de Software • Analista
Evaluación de cumplimiento de la implementación respecto al diseño.	Una vez finalizada la implementación de algún componente, al final del sprint se evalúa la conformidad de la implementación de dicho componente con respecto al diseño previamente evaluado. Esta actividad aplica hasta que la implementación cumpla con lo establecido en el diseño.	<ul style="list-style-type: none"> • Ingeniero de Software • Desarrollador
Evaluación de cumplimiento de la implementación respecto a los requerimientos.	Una vez finalizada la implementación de algún componente, al final del sprint se evalúa la conformidad de la implementación de dicho componente con respecto a los requerimientos del sistema de software. Esta actividad aplica hasta que la implementación cumpla con los requerimientos.	<ul style="list-style-type: none"> • Desarrollador • Analista
Construcción y revisión de los tests unitarios de integración y aceptación.	Las pruebas unitarias de integración y aceptación deben comprobar que las unidades del producto cumplan con las especificaciones. Esta actividad aplica hasta que la implementación supere las pruebas.	<ul style="list-style-type: none"> • Desarrollador • Verificador
Cumplimiento de estándar de codificación PEP8 Ejecución de análisis de código fuente para búsqueda de señales de baja calidad.	Utilización de herramientas de software para la comprobación del código y el aseguramiento del cumplimiento con el estándar de codificación PEP8. Aplica siempre que se haya escrito código en el transcurso del sprint.	<ul style="list-style-type: none"> • Desarrollador • Verificador
Ejecución de análisis de código fuente para búsqueda de señales de baja calidad.	Utilización de herramientas de software para la comprobación del código fuente para encontrar y corregir posibles causantes de baja calidad. Aplica siempre que se haya escrito código en el transcurso del sprint.	<ul style="list-style-type: none"> • Desarrollador • Verificador

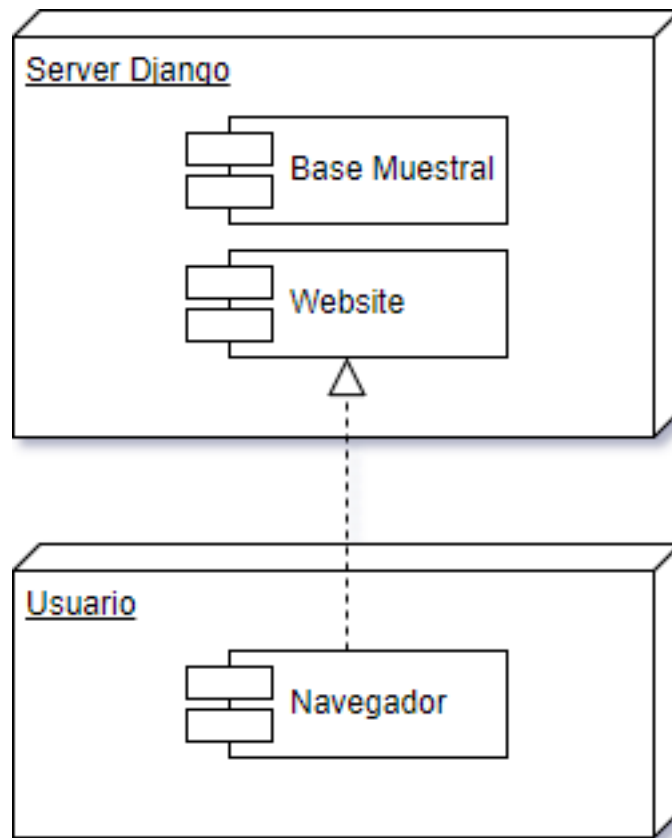
Posterior a realizar las actividades, se plantea para el siguiente sprint:

Posterior a	Realizar en el siguiente sprint
Actividad 1	Si el diseño no cumple con lo establecido en los requerimientos, en el siguiente sprint se deberá adecuar el diseño para que cumpla con dichos requerimientos
Actividad 2	Si la implementación de una o varias partes del sistema no cumple con el diseño, en el siguiente sprint se deberá adecuar dichas partes del sistema para que cumplan con lo establecido en el diseño.
Actividad 3	Si la implementación de una o varias partes del sistema no cumple con los requerimientos, en el siguiente sprint se deberá adecuar dichas partes para que cumplan con dichos requerimientos.
Actividad 4	Si una o varias unidades de código no pasan las pruebas unitarias, dichas unidades deberán ser adaptadas en el siguiente sprint.
Actividad 5	Si alguna parte del código no cumple con el estándar de codificación PEP8, deberá ser corregida de modo que cumpla con dicho estándar en el siguiente sprint.
Actividad 6	Si se identifican señales de baja calidad en el código, para el siguiente sprint estas deberán ser arregladas de modo que las pruebas muestren una mejora en la calidad.

7.4 Diagramas de UML

7.4.1 Diagrama de componentes

A continuación se muestra el diagrama de componentes del sistema.



7.4.2 Diagrama de clases

Diagrama de clases propio del sistema, que muestra el modelo a seguir para la resolución del reto de software que presenta el proyecto.

