Escuela de Ingeniería en Computación

Aseguramiento de la calidad del software

Profesor:
Saúl Calderón Ramírez.

SYRS Actualizado.

Erick Alfaro Hernández
2014098673
Ariana Bermudez Venegas.
2015019596
Ximena Bolaños Fonseca.
2015073844
Nicol Morice Sandí.
2015086588

19 de octubre del 2017

Contenidos

1 Especificación de los Requerimientos del Sistema

1.1 Propósito del sistema

El propósito del sistema consisten en la identificación automática de rostros en los principales puntos de acceso del país (el aeropuerto Internacional Juan Santamaría, el paso por Peñas Blancas, Paso Canoas y Sixaola), pensado para ser la futura implementación de un pasaporte electrónico. Se guardan 10 imágenes por persona para su futura identificación. Estas imágenes serán almacenadas en un banco de imágenes digitales de rostros de tosos los usuarios del sistema. El sistema identificará una imagen de rostro a partir del conocimiento inferido de este banco. Este software es desarrollado por la empresa BiomeSys, dedicada a la realización de sistemas biométricos.

Este sistema biométrico busca confiabilidad, eficiencia y precisión. Para implementar el sistema se usará el Análisis de Componentes Principales (ACP), que es una técnica estadística donde se elimina la información poco relevante y se conserva la más importante.

1.2 Alcance del Sistema

El sistema implementado será para reconocer caras y se llama ReconoceME. Este sistema tiene dos modos: el modo de entrenamiento y el modo de reconocimiento, la primera utiliza una base de datos muestral y la segunda recibe imagenes y devuelve quien ha detectado. De la aplicación se espera que ayude en los tramites de migración en los puestos fronterisos y aeropuertos, a que sean más rápidos y no sean fáciles de engañar como podría suceder con personal humano. Y como principal objetivo se tiene mejorar la seguridad del país.

1.3 Visión del Sistema

1.3.1 Contexto del Sistema

Actualmente los aeropuertos y fronteras manejan muchas personas, y las personas de migración pueden ser víctimas de personas que intentan engañar el sistema de revisión, o tal vez no reconocen los criminales internacionales que buscan huir a países como el nuestro. El sistema sería utilizado por los lugares previamente comentados, ayudando al personal y comparando sus datos para identificar criminales, robos de identidad, o uso de una identidad falsa.

1.3.2 Funciones del Sistema

A continuación se presentará las principales funciones que pueden llevar a cabo el sistema, también indicando sus condiciones y restricciones.
Funciones:

- Entrenar un sistema con muestras de sujetos para su posterior uso.
- Detectar caras a partir de su entrenamiento y mostrar a quien corresponde.
- Poder entrenar de nuevo al sistema con nuevas muestras.

Condiciones:

- Debe indicar a quien corresponde la fotografía si el sistema ha recibido las muestras del sujeto a quien se quiere identificar.
- Debe permitir volver a entrenar el sistema, lo cual implica volver a meter todas las muestras de nuevo.

Restricciones:

- No se puede identificar personas si las fotografías se encuentran mostrando su perfil, o ángulos diferentes a estar viendo directamente a la cámara.
- Y como anteriormente fue mencionado, no se puede detectar caras si el sujeto no tiene muestras en el modo de entrenamiento.

1.3.3 Características de Usuario

Se identificaron cuatro tipos de usuarios, no solamente los finales, estos son:

- Desarrolladores → crean el sistema → hacen el mejor esfuerzo para que cumpla con los requerimientos funcionales y no funcionales.
- De mantenimiento (en caso de que la aplicacion falle) → de la empresa, son contactados en caso de que el sistema ya siendo utilizado presente algún fallo mayor y no permita que sea utilizado en los puntos establecidos.
- Operadores → oficiales de migración → son los usuarios finales, ellos van a utilizarlo. Ellos esperan que el sistema sea fácil de usar para que no necesiten de muchas capacitaciones y además se sientan cómodos, así siguen utilizándolo. A futuro puede que este usuario desaparezca, siendo sustituido por el sistema.
- Viajeros → Es la persona que va al aeropuerto para realizar un viaje. Entre sus principales características se encuentra que no harán uso directo del sistema, para ello están los operadores. Son usuarios que esperan respuesta rápida, ya que pueden estar esperando un vuelo y pronto necesite abordarlo. También esperan que el proceso no sea engorroso, sino sencillo.

1.4 Requerimientos Funcionales

Para efectos de este programa y el uso que se le dará para el control de accesos al país mediante un pasaporte digital, se identifican las siguientes funcionalidades esenciales:

Requerimientos Funcionales		
Requerimiento	Prioridad	Explicación
REQ-001:Reconocer rostros con exactitud	Alta	Es de vital importancia que los rostros se puedan identificar con un gran nivel de exactitud.
REQ-002:Control de acceso	Media	El sistema manejara información sensitiva de muchas personas, de manera que se debe evitar que aquellos que no estén autorizados puedan utilizarla.
REQ-003:Cargar imágenes	Media	Cargar un conjunto de imágenes almacenada en la 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.
REQ-004:Entrenar el sistema	Alta	Entrenar el sistema, generando las 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.
REQ-005: Cargar un conjunto de muestras	Media	Cargar un conjunto de muestras de prueba, para medir la precisión del sistema.

1.5 Requerimientos de Usabilidad

Se estima que el sistema se estará utilizando con frecuencia y, además, con gran cantidad de personas que entran y salen del país, de manera que para asegurar que el proceso sea ágil el sistema será:

- **Fácil de entender:** se busca que el usuario pueda aprender a usarlo de manera sencilla para que se pueda implementar con mayor rapidez y se disminuya el tiempo de entrenamiento.
- **Fácil de operar:** con el objetivo de agilizar el tramite, los procesos para llevar a cabo la identificación de sujetos deben ser sencillos.
- **Fácil de aprender:** con el objetivo de agilizar el tramite, se busca que los encargados del manejo del sistema aprendan su funcionamiento facilmente.

1.6 Requerimientos de Rendimiento

El principal objetivo del sistema es la identificación de sujetos, de manera que se quiere un buen rendimiento en dicho aspecto. Para saber que el proyecto tenga un nivel alto de rendimiento, se va a realizar métricas de rendimiento. Entre dichas métricas, se encuentra la de sesión para poder supervisar las sesiones del usuario y así poder ver la tendencia del sistema. También se realizarán métricas de solicitud, para supervisar el volumen de las solicitudes, los tiempos de respuesta y los tiempos de procesamiento. Así entonces se busca que el sistema posea:

- **Comportamiento temporal:** se refiere al tiempo que le toma al sistema el identificar un rostro. Se busca que esté un rango entre 30 y 60 segundos.
- **Baja utilización de recursos:** no se quiere hacer un uso excesivo de la memoria, de forma que no se permite el utilizar más de 5 GB de la misma.

1.7 Interfaces del Sistema

Se cuenta con tres elementos esenciales como interfaces del sistema, los cuales se presentan a continuación:

- **Diagrama de clases:** representa la arquitectura interna del sistema y como sus distintos componentes se relacionan entre si. Se encuentra dividido basado en la forma MVC (Model, View, Controller). El mismo se puede consultar en la sección 3 - Arquitectura del sistema.
- **Elemento humano:** el sistema sera principalmente manipulado por una persona a cargo; que se encargará de supervisar el proceso y de guiar al sistema en las acciones que debe tomar.
- **Filestack:** es un sistema externo en donde se estarán almacenado las imágenes de los sujetos, que luego serán extraídas por el sistema para su entrenamiento y el reconocimiento. Dicha conexión se realizará mediante el API que el mismo Filestack provee para Python.

1.8 Operaciones del Sistema

1.8.1 Requerimientos de Integración Humano-Sistema

En este proyecto debemos de integrar las consideraciones humanas en el desarrollo del sistema, su diseño y en la gestión del ciclo de vida para de esta manera poder tener un rendiminto total de dicho sistema.
Para poder escribir estos requerimientos debemos considerar los atributos de calidad y asegurar que se están desarrollando de manera que incluyan capacidades y limitaciones humanas que pueden ser asociadas a estos atributos para la verificación en las actividades de evaluación del sistema. Entre estos atributos,

se consideró la usabilidad, la aptitud operativa y la sostenibilidad. Requerimientos tomados en consideración:

- Los operadores deben de tener un nivel alto de conocimiento del sistema, de tal manera que sepan como operarlo correctamente, darle mantenibilidad básica en caso de emergencia y darle un correcto uso.
- El personal de mantenimiento debe estar altamente capacitado para poder poner de nuevo el sistema en un buen funcionamiento en una cantidad de tiempo prudente.
- Los desarrolladores deben de poder dar un correcto mantenimiento al sistema y esto de manera eficiente.
- Los operadores deben de poder utilizar el sistema día y noche.
- Las personas que desean ingresar a Costa Rica deben de tener una previa breve explicación del sistema.
- El sistema debe ser utilizado solo por personal autorizado.

1.8.2 Mantenibilidad

En el aspecto de mantenibilidad, los desarrolladores han utilizado herramientas y técnicas para mejorar el código y así ayudar a quienes vayan a trabajar sobre la aplicación a futuro.
Para especificarlo cuantitativamente para arreglar problemas como pulgas encontradas en el sistema puede tomar entre 1 a 6 horas. En tiempo de ejecución, realizar algún cálculo de la matriz de covarianza, puede tomar de 2 a 5 minutos. Sin embargo, habrá una mejora para que tenga una ejecución más rápida, lo cual ayudará a la mantenibilidad, ya que al realizar cambios y pruebas, tomará menos tiempo.
(Actualización 16 de Octubre - 2017) El cálculo de la matriz de covarianza ya fue mejorado, por lo que toma menos tiempo entrenar el sistema, brindando la eficiencia que se había mencionado.
En cuestión de hablidad técnica, los lenguajes de programación seleccionados para desarrollar el sistema son de los más conocidos por la comunidad de programadores, por lo que no se necesitan de expertos difíles de encontrar. Al contar con un sistema documentado, también se les facilita a los próximos desarrolladores que van a utilizar y continuar con el mantenimiento de la aplicación. El programa cuenta con divisiones para la parte que sea lógica que es principalemente manejado por Python, y la parte web, la cual contiene código HTML, Javascript y CSS.

1.8.3 Confiabilidad

El sistema se considera critico para las labores de migración del país en sus distintos puntos de acceso, por lo que se debe contar con una alta confiabilidad

del mismo, tanto en aspectos como el reconocimiento de rostros, cuyo nivel de debe estar arriba del 97%; como en seguridad, estabilidad y mantenimiento de los datos de todos los sujetos registrados en el sistema.

Debido a lo anterior se especifican dos puntos, de fiabilidad que se encuentran relacionados con el cumplimiento de la confiabilidad, bajo los cuales el sistema debe comportarse de la manera adecuada para asegurar la continuidad de operación del mismo correctamente:

- **Tolerancia a fallos:** de suma importancia es que aunque se de un fallo, el sistema puede seguir operando al menos a un nivel mínimo establecido, en caso de un fallo a nivel general, que se debe poder mantener activo un 99.99% del tiempo. O bien si se da en uno de los puntos de acceso, este no debería de afectar a los otros puntos del país y quedar solo aislado a su entorno, estos deben mantenerse activos al menos el 95% del tiempo.
- **Recuperabilidad:** si lo ocurrido es una falla critica del sistema, debe ser posible que el mismo se pueda restablecer al nivel que tenía previamente, de preferencia en el menor tiempo posible y, además, sin que los datos del mismo se vean afectados por esta, es decir que la totalidad (100%) de los datos deben poder ser recuperados de manera concisa.

Debido a lo sensitivo e importante del sistema para el uso en el país, este es uno de los aspectos más importantes y de los que se debe garantizar su mayor cumplimiento para un buen desempeño del mismo.

1.9 Modos y Estados del Sistema

Para el cumplimiento de los objetivos del sistema se establecen dos estados esenciales de operación para el mismo y que se explicarán a continuación.

1.9.1 Entrenamiento

Ya que el sistema reconocerá rostros, debe entrenarse para lo mismo con el uso de diferentes muestras; el mismo tiene que llevarse a cabo adecuadamente para que cumpla con lo establecido (97% de confiabilidad en el reconocimiento). Por lo normal esto será un estado recurrente ya que se estarán agregando nuevos sujetos y debe entrenarse para reconocerlos.

1.9.2 Reconocimiento

Bajo este estado es con el que se espera que opere en los puntos de acceso del país, que como su nombre lo indica simplemente se encargara de reconocer rostros y en caso que el sujeto identificado sea buscado, se dará una alerta al operador. En una futura versión podría implementarse un mecanismo que le permita al sistema mejorarse a si mismo en este modo, como si estuviera en entrenamiento, pero este punto está fuera del alcance de esta versión y solamente se propone como un característica con la que podría contar a futuro.

1.10 Características Físicas

1.10.1 Requerimientos Físicos

El sistema tiene que trabajar en un lugar que pueda almacenar todas las muestras de los costarricenses y trabajar en conjunto con otros países, o solo tener en cuenta las personas que están siendo buscadas en otros paiśes, lo cual implica mucha memoria física. El programa cliente puede ser utilizado en cualquier computadora que contenga un sistema operativo como Windows, o Linux, reciente, y por ello, si la máquina deja de servir, se puede reemplazar fácilmente.

1.10.2 Requerimientos de Adaptabilidad

Con el tiempo la aplicación se puede implementar en lo que son la identificación de rostros en cuestiones médicas,registros nacionales por lo que se puede llegar y ampliar a sistemas dedicados. Por lo que el sistema no se puede encerrar simplemente en aerolíneas por lo que el sistema debe poder ser empaquetado para poder utilizarse en diferentes códigos y sistemas.

1.11 Condiciones Ambientales

La fiabilidad de nuestro sistema depende de un entorno estable.Por lo que debemos asegurarnos que el sistema funcione de una manera fiable mientras permanece dentro del rango de sus especificaciones operativas. Aspectos a considerar:

- El sistema debe soportar los constantes rayos X
- El sistema debe soportar golpes constantes de las maletas o personas
- El sistema debe de estar en un ambiente con un rango de tempratura de 21 a 23oC.(70 a 74 oF)
- El sistema debe de estar favorablemente en un ambiente con un nivel de humedad entre 45% y 50%, de esta manera evitar problemas en el sistema y evitar la corrosión del hardware.
- El sistema no deberá variar la humedad más del 10% por hora de funcionamiento.
- El sistema no debe de publicar ningún contenido por cuestiones legales sobre sus pasajeros y económicamente se debe de estar invirtiendo en el constante mantenimiento de esta.

1.12 Seguridad del Sistema

El sistema como en todo debe de tener seguridad, entre las cuales la restricción de las personas en lo que son todos los datos de la persona y como lo que son los fotos de las personas, número de pasaporte, cédula y nombres completos. Para

lo cuál se va a tener un protocolo de seguridad muy fuerte para que personas maliciosas o ajenas al aeropuerto como ladrones de identidad o bien hackers que quieran alterar el reconocimiento facial para permitir la ida y venida de personas ilícitas al país. Por ello es importante que muy poca gente tenga acceso al código o bien a la base de datos muestral. Para ello se deberá hacer la implementación de cifrados y logins para que no sucedan incovenientes.

1.13 Administstracción de la Información

El sistema va a hacer manejo más que todo para el manejo de imágenes por lo cuál va a haber una constante carga de datos de lo que son las imágenes que serán procesadas como un FileStack, a través de un API. Se intentará que el sistema no consuma más cantidad de datos de la cuenta por medio de algoritmos que aumentan la eficiencia y la cantidad de almacenamiento.

1.14 Políticas y Regulaciones

El sistema debe de ser multilingüe ya que será utilizado en un aeropuerto y personas de todo el mundo lo utilizarán constantemente para la verificación de sus rostros y necesitan entender su procedimiento. El sistema de soportar estar cerca de lo que son rayos X y sobre todo no permitir publicación alguna de los datos que fueron ingresados en el sistema. Cada 10 años se debe de estar actualizando la información de los individuos para que no haya ningún inconveniente a la hora de estar identificando los rostros de las personas.

1.15 Sostenibilidad del Ciclo del Vida del Sistema.

En el proyecto debemos realizar actividades, como la revisión, recopilación y análisis de medidas, para poder lograr calidad en nuestro sistema. Al igual que tener un control en el mantenimiento del ciclo de vida para poder proporcionar niveles operacionales, soporte, repuestos, siministros, documentación técnica, entre otras cualidades necesarias. Escoger la metoldología SCRUM para el desarrollo de este proyecto, nos ayuda en la mantenibilidad del sistema, al igual que el uso del estándar IEEE 730. Entre estas actividades se van a realizar ó se han realizado:

- Reunión de planación del proyecto.
- Reunión del alcance del proyecto.
- Reunión de la estimación del proyecto.
- Uso de herramientas ALM.
- Planeamiento del proceso de ACS.
- Aseguramiento del producto.
- Aseguramiento del proceso.

1.16 Empaque, manipulación, envío y transporte

Esta sección busca dar a conocer como se llevará a cabo el manejo de los paquetes y la manipulación del sistema a la hora de ser entregado y distribuido a los diferentes puntos del país:

- El procesamiento de empaquetado debe de ser realizado por un desarrollador, junto a un supervisor.
- La manipulación del sistema debe ser elaborada solamente por los desarrolladores. Otras partes interesadas no deben de poder manipular el sistema.
- El sistema se empaquetará junto con el código fuente y toda la documentación realizada. Esto en un único archivo comprimido.
- Se dará el sistema únicamente a los puntos de acceso del país.

1.17 Verificación

Los métodos de verificación de requerimientos están definidos de la siguiente manera:

- Inspección: tener un control de las implementaciones o instalaciones de componentes.
- Análisis: Verificación basado en datos recolectados de pruebas que verifican la funcionalidad.
- Demostración: Verificación sin métodos cuantitativos. Implica que el requerimiento está completo.
- Prueba: Verificación con características cuantitativas con medidas cuantitativas.

Verificación		
ID Requerimiento	Método	Requerimiento
REQ-001	Prueba	Reconocer rostros con exactitud
REQ-002	Demostración	Control de acceso
REQ-003	Prueba	Cargar imágenes
REQ-004	Demostración	Entrenamiento del sistema
REQ-005	Prueba	Cargar un conjunto de muestras

1.18 Suposiciones y Dependencias

Una suposición sobre el producto es que siempre será utilizado en cualquier computadora que contenga un sistema operativo como Windows, o Linux.
Se asume que la computadora tiene una capacidad alta para poder ejecutar bien el sistema sin ningún tipo de problema.
Se asume que la computadora cuenta con una cámara de buena calidad para un mejor funcionamiento al reconocer los rostros faciales.
Se asume que este sistema solamente será utilizado por la seguridad en los puntos de acceso al país.

1.19 Validación de diseño

Tabla Validación de diseño

Ítem de diseño	Requerimiento	Explicación
Person	REQ-001	Para poder identificar a la persona al hacer el proceso, se debe de verificar con la información de Person en el sistema. Para identificarla debe de estar su información en esta clase.
Sample	REQ-004	Para poder identificar a la persona al hacer el proceso, se deben ver o tener las muestras de dicha persona, así poder hacer la proyeccíon.
System	REQ-004, REQ-001	En System tiene los métodos de aprender, que es cuando se entrena el sistema y también tiene el de reconocer, que es cuando se carga la foto y se identifica a quién pertenece.
FacadeLearning	REQ-004	Conoce la clase del subsistema que es responsables de la petición de aprendizaje.
DAOFilestack	REQ-003	Es la clase encargada de poder darnos el conjunto de imágenes que se cargaron, las cuales estaban en una dirección prevista.
FacadeRecognition	REQ-001	Conoce la clase del subsistema que es responsables de la petición de reconocimiento.

UIRecognition	REQ-001	Implementa la funcionalidad de reconocimiento. Realiza el trabajo solicitado por FacadeRecognition.
UILearning	REQ-004	Implementa la funcionalidad de aprendizaje. Realiza el trabajo solicitado por FacadeLearning.
ViewLearning	REQ-004	Clase de vista que va a permitir hacer el uso del método de aprendizaje.
ViewRecognition	REQ-001	Clase de vista que va a permitir hacer el uso del método de reconocimiento.

(Actualización de 16 de octubre del 2017) El equipo de desarrollo, junto al administrador de la configuración llevaron a cabo cambios en el modelo porque mientras se iba desarrollando el sistema, se dieron cuenta que el entrenamiento y el reconocimiento comparten muchas de las funciones, es por ello que se unificaron las clases FacadeRecognition y FacadeLearning, lo cual lleva a unir UIRecognition y UILearning, por lo que finalmente implica junta ViewLearning y ViewRecognition. Por ello en la tabla anterior están de color amarillo. También DAOFilestack se removió de las clases, ya que no es necesaria generar una clase solo para ello.

2 Estándar de Codificación

Mantener un buen estándar para el código escrito y para todos los miembros del grupo es de suma importancia, ya que se dice que el código es más frecuentemente leído que escrito. Por lo que es muy importante que se puede comprender correctamente y evitar mal entendidos del mismo, incluso entre aquellos que no fueron parte de la creación del código.

Con esto en mente hemos optado por escoger el estándar de codificación PEP8, que es una guía de estilo especifica para Python que nos permita mantener una consistencia en todo el código escrito durante la realización del proyecto.

2.1 Estándar PEP 8

Viene a ser una guía más de codificación y como muchas de estas guías, lo que busca es mejorar la legibilidad del código, así como mantenerlo consistente para todo el proyecto y entre todos los participantes. Este último punto es muy importante y la misma guía de PEP8 así lo establece ("A style guide is about consistency.").

Sin embargo, nos destaca también la importancia de saber cuando ser inconsistente, dándonos como ejemplo el que no deberíamos romper la retrocom-

patibilidad solo para cumplir con la guía; también nos presenta otros ejemplos pero serán omitidos aquí.

Está es solo una breve introducción de los sería el estándar PEP8 para codificación en Python, si se desea saber más a fondo sobre el se puede visitar su página oficial, como se encuentra en [1].

2.2 ¿Por qué PEP 8?

Esencialmente se escogió este estándar porque la manera en la que establece la escritura del código es muy similar a la manera en la que ya se trabajo durante la realización del POC, solamente con unos pequeños ajustes.

Por lo cual se consideró que seria un ahorro de tiempo ya que no deberíamos utilizar demasiado tiempo para aprender a seguir este estándar, seria más sencillo acostumbrase a lo que establece y seria menos forzado el cambio y ajuste, porque como se menciono anteriormente es similar a como se trabajó en la etapa anterior.

Aunado a esto se tiene que el mismo ya se encuentra incluido en el plugin PyDev de Eclipse, ahorrando tiempo de instalación y configuración de la herramienta, disminuyéndolo a básicamente solo activarla.

2.3 Herramienta para la verficación del estándar

Como se mencionó anteriormente, la herramienta ya se encuentra incluida en el plugin PyDev de Eclipse y es necesario solo activarla y señalar la manera en la que se quiere que se muestre cuando no se está cumpliendo el estándar (como advertencias, el escogido en nuestro caso; errores o información).

Luego que se activa, la misma estará revisando cada vez que se guarde el archivo u archivos sobre los que se está trabajando y mostrará las lineas donde no se esté cumpliendo, así como una descripción de lo que está mal; lo cual nos pareció muy útil, sencillo y rápido.

2.4 Conclusiones

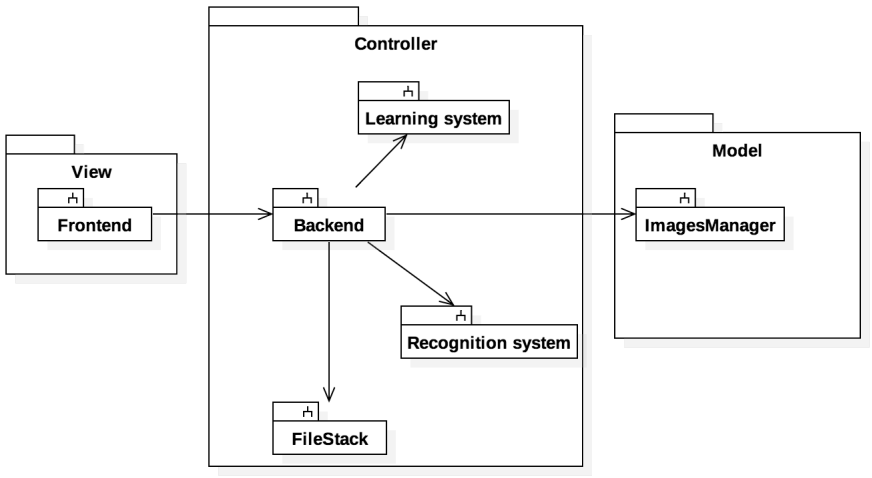
PEP 8 es un buen estándar de codificación, ayuda a mantener el código conciso y a aumentar la legibilidad, además, es similar a como se trabajó para el POC, disminuyendo tiempo de aprendizaje, de implementación e incluso la adaptación del código ya existente al mismo. Agregado al hecho de su fácil activación y utilización la hace la guía ideal para seguir durante el resto del proyecto.

3 Arquitectura del Sistema

3.1 Diagrama de componentes

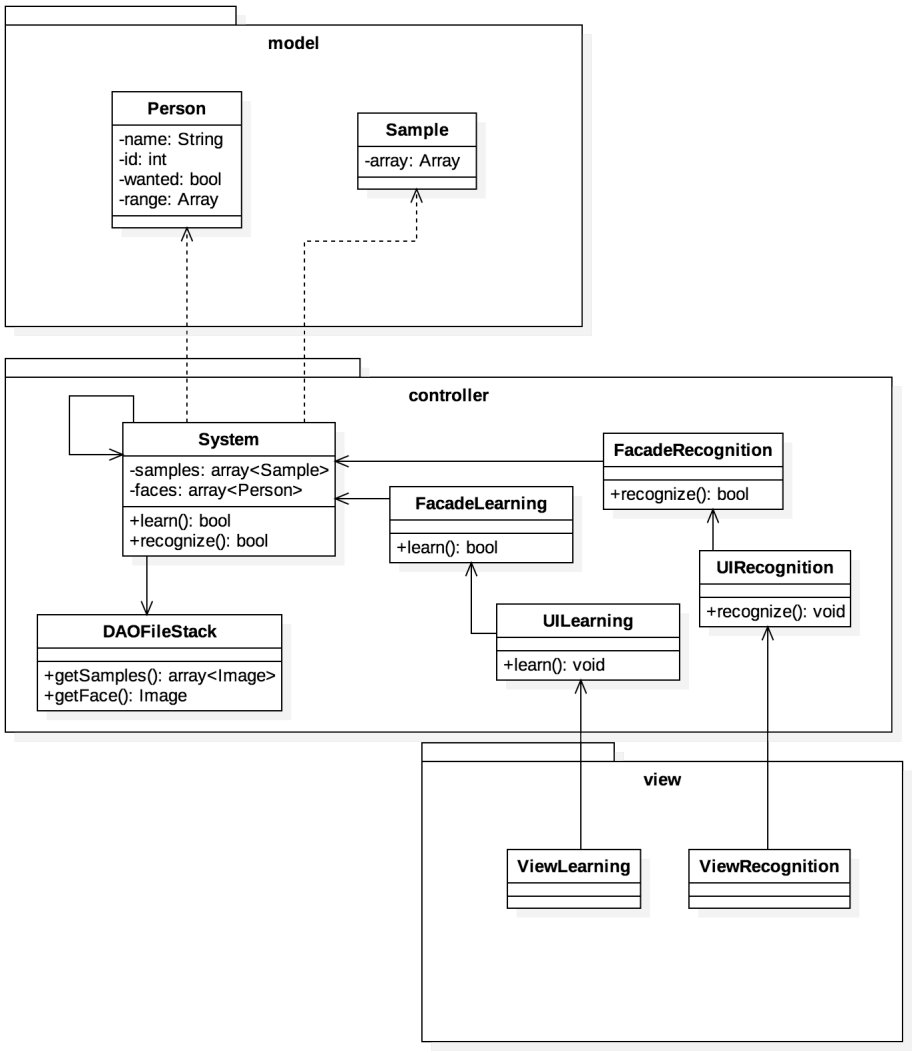
Este diagrama presenta una visión general sobre la estructura que tendrá el sistema, así como algunos de sus más importantes componentes. Para este caso

particular se modelado de forma que coincida con el diseño MVC (Model, View, Controller).

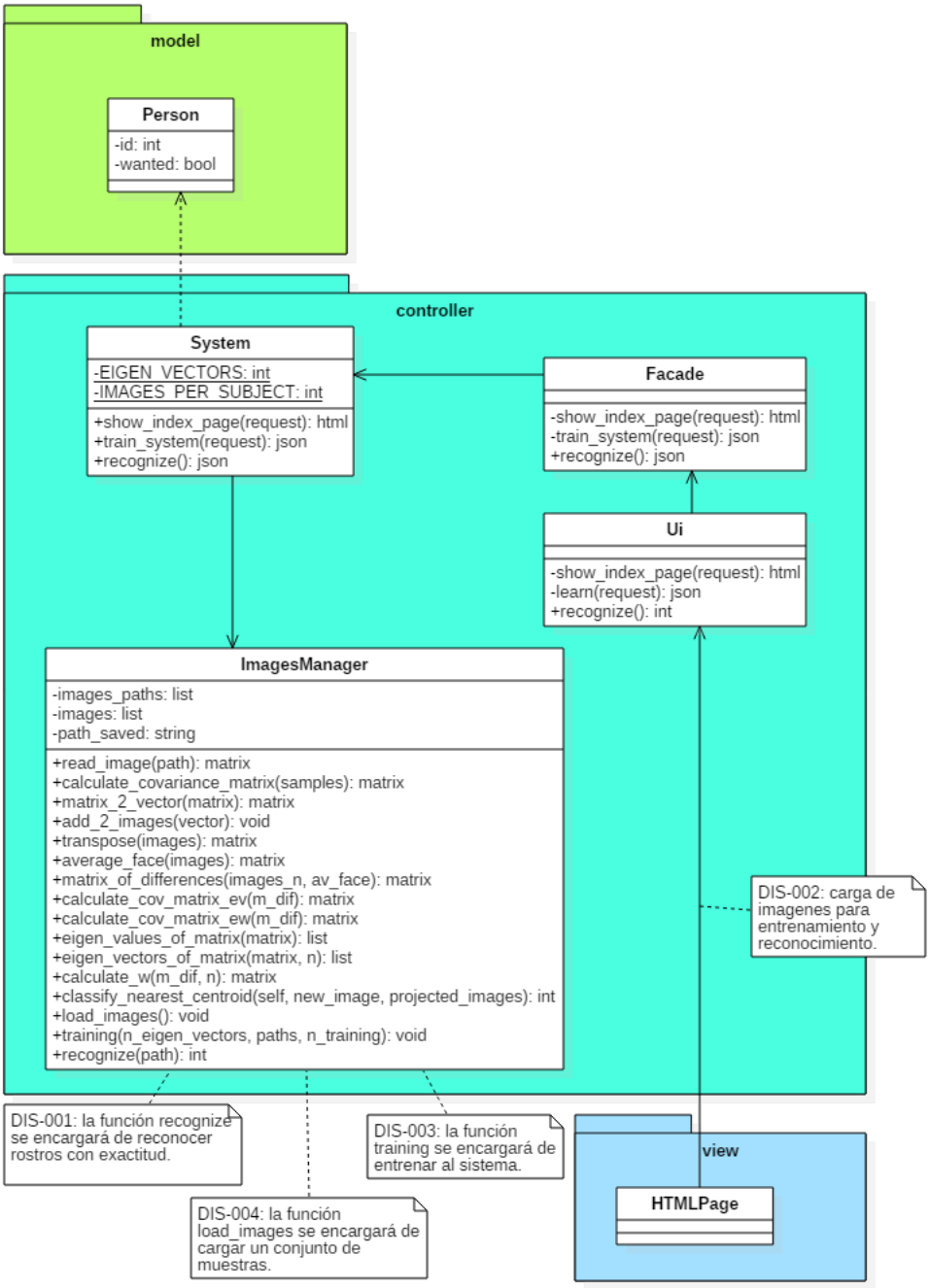


3.2 Diagrama de clases

Diagrama que representa la forma en la que se relacionan todas las diferentes clases del sistema, ya que el mismo se está modelo con orientación a objetos. Por orden aquí también se plantea el disenõ MVC.

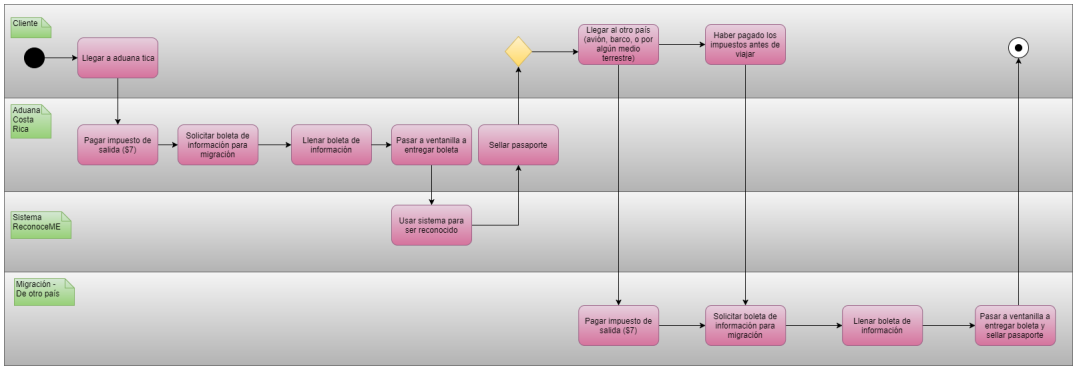


(Actualización del 16 de Octubre, del 2017) A continuación mostraremos el modelo actualizado. Los cambios se realizaron ya que el equipo de desarrollo junto al administrador de la configuración consideraron necesario, y el modelo anterior no era práctico. Más adelante se detallara más la información de estos cambios.


model
Person
-id: int
-wanted: bool
controller
System
-EIGEN_VECTORS: int
-IMAGES_PER_SUBJECT: int
+show_index_page(request): html
+train_system(request): json
+recognize(): json
Facade
-show_index_page(request): html
-train_system(request): json
+recognize(): json
Ui
-show_index_page(request): html
-learn(request): json
+recognize(): int
ImagesManager
-images_paths: list
-images: list
-path_saved: string
+read_image(path): matrix
+calculate_covariance_matrix(samples): matrix
+matrix_2_vector(matrix): matrix
+add_2_images(vector): void
+transpose(images): matrix
+average_face(images): matrix
+matrix_of_differences(images_n, av_face): matrix
+calculate_cov_matrix_ev(m_dif): matrix
+calculate_cov_matrix_ew(m_dif): matrix
+eigen_values_of_matrix(matrix): list
+eigen_vectors_of_matrix(matrix, n): list
+calculate_w(m_dif, n): matrix
+classify_nearest_centroid(self, new_image, projected_images): int
+load_images(): void
+training(n_eigen_vectors, paths, n_training): void
+recognize(path): int
DIS-002: carga de imagenes para entrenamiento y reconocimiento.
DIS-001: la función recognize se encargará de reconocer rostros con exactitud.
DIS-003: la función training se encargará de entrenar al sistema.
DIS-004: la función load_images se encargará de cargar un conjunto de muestras.
view
HTMLPage

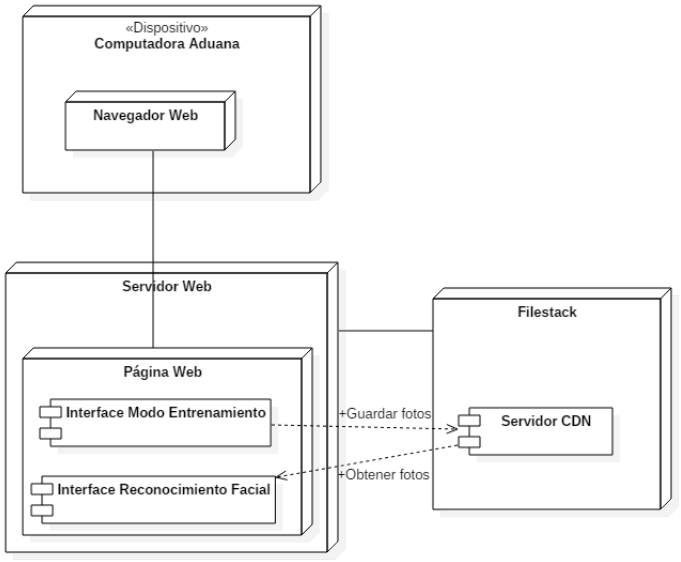
3.3 Diagrama de negocios

Este diagrama muestra el proceso que implica salir del país ya sea por medio de aeropuertos, barcos o de manera terrestre. Siempre se tiene que seguir este proceso y también muestra en que momento se usaría.



3.4 Diagrama de despliegue

Este diagrama nos ayuda a poder ver las relaciones físicas entre los componentes hardware y software del sistema de reconocimiento facial.



4 Actividades a realizar

Como manera de mantener una mejor organización y estar consistentes en lo que respecta al desarrollo del sistema, se plantearon un serie de actividades (conocidas como sprints) que nos permitan cumplir con dichos objetivos. Asimismo, contar con una guía de lo que se debe hacer, lo que se ha hecho y lo que esta en progreso; sin mencionar que ayudan significativamente a asignar el trabajo de manera más relevante a los miembros del grupo.

Se puede denotar que la etapa de Análisis abarca los Sprints 1, 2 y parte del 3; la etapa del Diseño el Sprint 3; y por último la Implementación y las Pruebas se dan en el Sprint 4 y 5. Sin embargo, puede que algunas de estas etapas se encuentren también en otros sprints además de los que corresponde.

4.1 Sprint 1

Este Sprint consiste en la etapa de elicitación y análisis del proyecto. Esta etapa permite saber si el prototipo y la idea del proyecto si se están comprendiendo de la manera adecuada, además de que la empresa que pidió el software permita a la empresa desarrolladora ser contratada.

1. **Definir**
   Responsables: Todos los miembros del equipo
   Tiempo: 2 días
   Se debe realizar la definición de los roles y actividades para cada una de las tareas correspondientes a esta etapa inicial.

2. **Trozos de Código**
   Responsables: A cada uno de los miembros les toco un trozo.
   Tiempo: 5 días cada uno
   Se deben de realizar cuatro trozos de código como base, de cierta manera prototipado para lo que va a ser el proyecto en general.

3. **Documentación**
   Responsables: A todos los miembros del equipo.
   Tiempo: 17 días
   Se debe de realizar un documento POC para la explicación de las partes anteriores. Además los trozos de Código implementados deben de ser documentados por medio de Java-Doc.

4. **Unit-Test**
   Responsables: Todos los miembros del equipo.
   Tiempo: 3 días
   Se debe de comprobar que cada uno de los trozos implementados funcionaba de una manera adecuada. A partir de lo que son las la entrada y la salida.

4.2 Sprint 2

Esta etapa consiste en la definición de los atributos y métricas que se van a implementar en este proyecto. Además, las herramientas que van a permitir que se cumplan con estas.

1. **Definición de Atributos**
   Responsables: Todos los miembros del equipo.
   Tiempo: 11 días
   Se deben de definir los atributos que se van a utilizar para la definición de la calidad.

2. **Justificación**
   Responsables:Todos los miembros del equipo.
   Tiempo: 11 días
   El porqué se eligió ese atributo de calidad y como se va a implementar.

3. **Métrica, Nivel Requerido y Herramientas**
   Responsables: Todos los miembros del equipo.
   Tiempo: 11 días
   Se definen qué métricas se van a utilizar con base a los atributos, los niveles requeridos para cada una de estas y que herramientas se van a utilizar para la comprobación de cada una de estas.

4.3 Sprint 3

En esta etapa se realizará la elicitación de requerimientos más detalladamente y la diagramación de la estructura del sistema. También se plantearán las distintas actividades que realizarán para completarlo, así como sus respectivas verificaciones de calidad.

1. **Definición de Estándar de Codificación**
   Responsables: Ximena y Erick
   Tiempo: 11 días
   Se deberá investigar y escoger de manera justificada un estándar para la escritura de código, con el objetivo de mantener una legibilidad y consistencia en el mismo, durante toda la realización del proyecto.

2. **Creación del SyRs**
   Responsables: Todos los miembros del equipo.
   Tiempo: 11 días
   Plasmar de manera formal los diferentes requerimientos que se quieren para el sistema; así como otros varios aspectos del mismo que sean relevantes para el usuario.

3. **Definición de Actividades para el Aseguramiento de la Calidad**
   Responsables: Ximena y Erick
   Tiempo: 11 días

Ya que se quiere que el producto sea de calidad, se deberán definir distintas actividades a realizarse al final de cada sprint para asegurarse que el sistema esté cumpliendo con las métricas, objetivos y calidad planteados.

4. **Construcción de Diagramas**
   Responsables: Ariana, Erick y Nicol
   Tiempo: 11 días
   Para mostrar una mejor idea sobre la estructura del sistema, se elaborarán diagramas que ayuden a proporcionar una ayuda visual para el entendimiento del mismo.

4.4 Sprint 4

Este se efocará principalmente en la adaptación del sistema presentado en el POC para que logré cumplir tanto con las métricas definidas como con el estándar de codificación escogido. También se comenzará la implementación de otros trozos de código. Así se tiene que las sub-actividades a realizar son las siguientes.

1. **Cambiar algoritmo de entrenamiento/reconocimiento**
   Responsables: Erick y Ximena
   Tiempo: 11 días.
   Aunque ya se encuentra implementado un algoritmo para el entrenamiento del sistema, este no alcanza para cumplir con las métricas de rendimiento que se plantearon anteriormente. De forma que el primer paso será el codificar una forma reducida del algoritmo que permita el ahorrar tiempo del proceso.

2. **Adaptar los archivos existentes al estándar PEP 8**
   Responsables: Ariana
   Tiempo: 11 días
   Como se mencionó previamente, se utilizará el estándar de codificación PEP 8, de manera que para mantener la consistencia en el código del sistema los archivos que sean reutilizados deberán ser adaptados al mismo.

3. **Conexión con Filestack**
   Responsables: Nicol
   Tiempo: 3 días
   Debido a que las imágenes a procesar tanto para reconocimiento como para entrenamiento no serán almacenadas en el sistema, sino que se manejarán de manera externa es necesario codificar un modulo que permita la obtención de las imágenes que allí se almacenen para su posterior procesamiento.

4. **Documentación del sistema**
   Responsables: todos los miembros del equipo.
   Tiempo: 11 días.

Está será un actividad recurrente, ya que la idea es ir documentando el sistema conforme se va desarrollando (tanto documentación interna como externa) para la ayuda de otros que puedan llegar a trabajar en el sistema también o para el mismo equipo a corto y/o largo plazo.

5. **Evaluación de conformidad del diseño respecto a los requerimientos**
   Responsables: todos los miembros del equipo.
   Tiempo: 11 días.
   Se debe de realizar una revisión en el código de que cada una de las partes del sistema cumplan con cada uno de los requerimientos establecidos en este documento.

6. **Evaluación de conformidad de la implementación respecto al diseño**
   Responsables: todos los miembros del equipo.
   Tiempo: 11 días.
   Se debe de denotar que la estructura del código sea conforme se estableció en este documento, en especial el diagrama de clases.

7. **Cumplimiento de estándares**
   Responsables: todos los miembros del equipo.
   Tiempo: 11 días.
   Según se definió cada él estándar PEP 8 de python de debe de revisar a partir de eclipse el cumplimiento de este en el código.

4.5 Sprint 5

En este se planea la implementación concreta del reconocimiento de rostros, así como la mejora funcional y visual de la página web que se utilizará como medio para interactuar con con el sistema. También se mantendrá la idiologia que documentarlo conforma se realiza.

1. **Implementación del reconocimiento facial**
   Responsables: todos los miembros del equipo.
   Tiempo: 11 días.
   Aunque se realizará de manera similar al entrenamiento (utilizando el mismo algoritmo), aún no hay una implementación concreta para la realización de esta tarea, por lo que es una de las prioridades para este sprint.

2. **Mejora estética de la página web**
   Responsables: Nicol y Ximena
   Tiempo: 11 días.
   Debido al rápido desarrollo que se dio para el POC la página web no cuenta con diseño atractivo ni mucho menos intuitivo, por lo que mejorar su apariencia en esencial ya que ayudará a que el sistema sea más fácil de

entender y de utilizar; teniendo siempre en cuenta la usabilidad para el usuario.

3. **Mejora funcional de la página web**
   Responsables: Erick y Ariana
   Tiempo: 11 días.
   Si bien la página presentada en el POC era funcional en cierta medida, no contaba con las funciones necesarias y mucho menos desplegaba una respuesta correcta al usuario, así que esto se debe cambiar para coincidir con las funcionalidades planteadas anteriormente.

4. **Documentación del sistema**
   Responsables: todos los miembros del equipo.
   Tiempo: 11 días.
   De nuevo, esto es de vital importancia y como se menciono será un proceso recurrente, por lo que también será parte de esta y futuras actividades para mantener una buena consistencia durante todo el desarrollo del sistema; y el mismo sea más sencillo de comprender y de mantener.

5. **Construcción y revisión de los test unitarios, de integración y aceptación**
   Responsables: Ariana y Nicol
   Tiempo: 11 días.
   Revisión y construcción de cada uno de los test unitarios para las secciones importantes de implementadas en el código y verificar que cada parte realice bien su función.

Referencias

[1] Van Rossum, G., Warsaw, B. & Coghlan, N. (2001). *PEP 8 – Style Guide for Python Code.* [online] Python.org. Available at: https://www.python.org/dev/peps/pep-0008/ [Accessed 7 Sep. 2017].

[2] Alfaro E.,Bermúdez A.,Bolaños X & Morice N. (2017). *Tarea 1, Atributos de Calidad y Métricas.* Available at: https://www.overleaf.com/read/pbprhkhbzjzk

[3] Alfaro E.,Bermúdez A.,Bolaños X & Morice N. (2017).*DOCUMENTACIÓN DE AVANCE DE PROYECTO I.* Available at: https://www.overleaf.com/read/gmdpdrkfxfwf

[4] ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes –Requirements engineering," in ISO/IEC/IEEE 29148:2011(E) , vol., no., pp.1-94, Dec. 1 2011 doi: 10.1109/IEEESTD.2011.6146379

[5] IEEE Standard for Software Quality Assurance Processes," in IEEE Std 730-2014 (Revision of IEEE Std 730-2002) , vol., no., pp.1-138, June 13 2014 doi: 10.1109/IEEESTD.2014.6835311