

Proof of Concept (POC)

Aseguramiento de la Calidad del Software

Francisco Monge Zúñiga (2013029434), Student in Computer Engineering at Costa Rica Institute of Technology,
 Joseph Salazar Acuña (2015100516), Student in Computer Engineering at Costa Rica Institute of Technology and
 Andrés Gutiérrez Salas (201223823), Student in Computer Engineering at Costa Rica Institute of Technology

Abstract—El presente proyecto pretende ejecutar una prueba de concepto (POC = Proof Of Concept) en el marco de probar las tecnologías disponibles que puedan sustentar el desarrollo del proyecto del curso Aseguramiento de la Calidad del Software de la carrera de Ingeniería en Computación del Tecnológico de Costa Rica.

Index Terms—POC, proof of concept, Python, Django, Flask, pruebas unitarias, Keras, pesos, scrum, git, diagrama de componentes, back-end, front-end.

I. INTRODUCCIÓN

El presente proyecto pretende ejecutar una prueba de concepto (POC = Proof Of Concept) en el marco de probar las tecnologías disponibles que puedan sustentar el desarrollo del proyecto del curso Aseguramiento de la Calidad del Software de la carrera de Ingeniería en Computación del Tecnológico de Costa Rica.

Dicho proyecto consiste en utilizar tecnología web para implementar un sistema de segmentación de imágenes digitales para la detección de células de tejidos de glioblastoma (tejido cerebral canceroso) tomadas de un microscopio basado en fluorescencia.

Lo que se pretende es realizar una investigación y una serie de pruebas cortas que permitan determinar cuáles herramientas y software específicos se amoldan mejor a las necesidades del proyecto, de manera tal que a futuro no se tenga que estar tomando decisiones sin fundamento en las elección de dichas herramientas.

En las siguientes secciones se encuentra la especificación del alcance total del proyecto, el diagrama de componentes que refleja la estructura general del sistema a desarrollar, los análisis de los trozos de código implementados o las pruebas cortas y cuáles fueron los problemas encontrados durante la ejecución del proyecto y la solución que se propuso a los mismos. Por último encontrará una sección de anexos en la que se incluye información o enlaces para acceder a la implementación de algunas partes del proyecto.

II. ALCANCE DEL PROYECTO

El proyecto a desarrollar consta de varios objetivos que se desea cumplir. El objetivo principal es el desarrollo de una prueba de concepto que permita a los estudiantes enfrentarse a algunas de las herramientas disponibles para verificar cuáles de ellas son más accesibles y funcionales en

miras a lograr desarrollar con éxito el proyecto completo del curso. Además, dentro de las asignaciones requeridas se tiene:

- a) El uso de la metodología de trabajo Scrum, por medio de la herramienta gratuita Zoho.com.
- b) El uso de una herramientas de control de versiones como Git.
- c) Creación de un diagrama de componentes de alto nivel que permita reflejar, de manera general, cuáles son los componentes o clases necesarios para el sistema.
- d) Implementar al menos cuatro trozos de código funcionales.
- e) Generar la documentación interna del código a partir del estándar de Javadocs, por medio de una herramienta como Doxygen o similar.
- d) Generar la documentación general del POC, es decir, el presente documento.

III. DIAGRAMA DE COMPONENTES

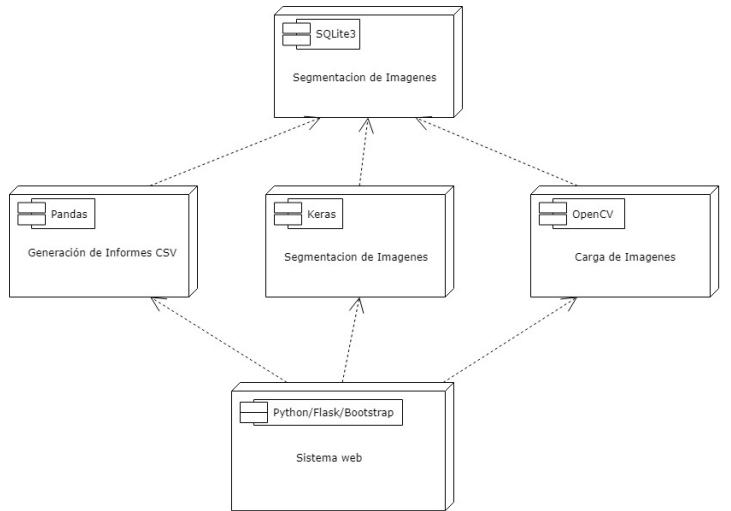


Fig. 1: Diagrama de componentes de alto nivel del sistema implementado.

IV. ANÁLISIS DE PIEZAS DE CÓDIGO

A. Pieza 1: Cargar modelo de red neuronal en Keras

Esta primera pieza consiste en poder cargar los pesos de entrada de un modelo sobre una red neuronal. Para poder cargar este modelo se creó primero una red neuronal sencilla,

que soporte la entrada de los datos. Posteriormente se opta por guardar el modelo completo para así probar la capacidad de cargar y, eventualmente, ejecutar un modelo funcional.

```
Console X
<terminated> LoadWeight.py [/Users/joshsalazar/anaconda3/bin/python3.6]
Using TensorFlow backend.
2018-08-23 22:58:00.117483: I tensorflow/core/platform/cpu_feature
2018-08-23 22:58:00.117807: I tensorflow/core/common_runtime/proce
32/768 [>.....] - ETA: 4s
768/768 [=====] - 0s 262us/step
acc: 77.86%
```

Fig. 2: Resultado de Cargar el Modelo en Keras

B. Pieza 2: Cargar un archivo .CSV con Pandas

Esta segunda pieza consiste en poder cargar archivos CSV mediante el framework llamado Pandas. Pandas es muy versátil a la hora de abrir y cargar archivos CSV permitiendo parametrizar el archivo.

```
Console X
<terminated> LoadCSV.py [/Users/joshsalazar/anaconda3/bin/python3.6]
('csv_file': Unnamed: 0 first_name last_name age preTestScore postTestScore
0 0 Jason Miller 42 4 25,000
1 1 Molly Jacobson 52 24 94,000
2 2 Tina . 36 31 57
3 3 Jake Milner 24 . 62
4 4 Amy Cooze 73 . 70, '_name': 'csv1.csv'}
<class 'pandas.core.frame.DataFrame'>
('csv_file': Region Total Profit
0 Australia and Oceania ... 951410.50
1 Central America and the Caribbean ... 248406.36
2 Europe ... 224598.75
3 Sub-Saharan Africa ... 19525.82
4 Sub-Saharan Africa ... 639077.50
5 Australia and Oceania ... 285087.64
6 Sub-Saharan Africa ... 693911.51
7 Sub-Saharan Africa ... 510216.66
8 Sub-Saharan Africa ... 152114.20
9 Sub-Saharan Africa ... 584073.87
10 Asia ... 7828.12
11 Sub-Saharan Africa ... 306037.92
12 Asia ... 606334.72
13 Central America and the Caribbean ... 1487320.02
14 Asia ... 122819.06
15 Europe ... 122865.12
16 Asia ... 1208744.24
17 Sub-Saharan Africa ... 85033.80
18 Asia ... 634745.90
19 Australia and Oceania ... 337937.60
20 Europe ... 714157.00
21 Europe ... 122029.78
22 Central America and the Caribbean ... 122686.50
23 Australia and Oceania ... 5270.67
24 Europe ... 127054.20
25 Europe ... 315574.05
26 Australia and Oceania ... 130899.18
27 Sub-Saharan Africa ... 14831.02
28 Europe ... 80241.84
```

Fig. 3: Resultado de Cargar el CSV en Pandas

C. Pieza 3: Cargar imágenes con OpenCV

Esta tercer pieza consiste en utilizar la librería OpenCV, que tiene funcionalidades muy efectivas y potentes que permiten cargar imágenes de manera sencilla.

D. Pieza 4: Elegir y mostrar imágenes dinámicamente en página web

Esta cuarta pieza consiste en brindar al usuario de la página web la posibilidad de elegir las imágenes que desee (una o varias) de cualquier directorio de la computadora. Una vez que los archivos han sido elegidos, los mismos deben ser enviados al back-end, de manera tal que este pueda disponer de ellos. En esto caso específico lo que el sistema hace es mostrar

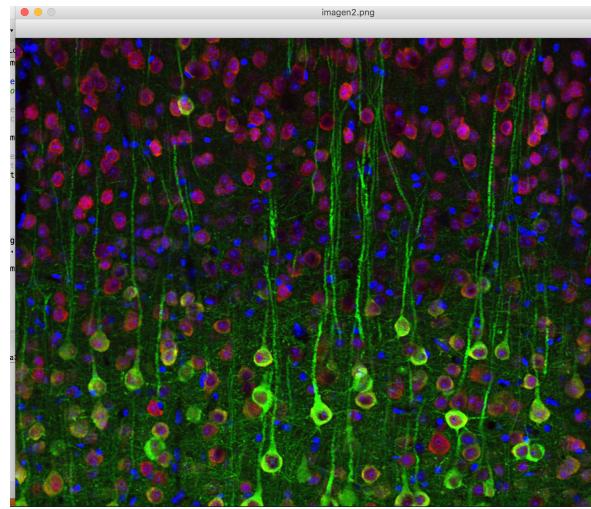


Fig. 4: Resultado de cargar imágenes en OpenCV

al usuario la serie de imágenes que fueron elegidas, esto en varios formatos. Se implementó un carrusel, que permite visualizar las imágenes sin necesidad de oprimir botones para desplazarse. También se implementó una galería de thumbnails para facilitar la visualización de la carga completa. Todo esto se genera de manera dinámica, es decir, el usuario puede seguir eligiendo imágenes cuantas veces desee y las mismas serán desplegadas en la página web.

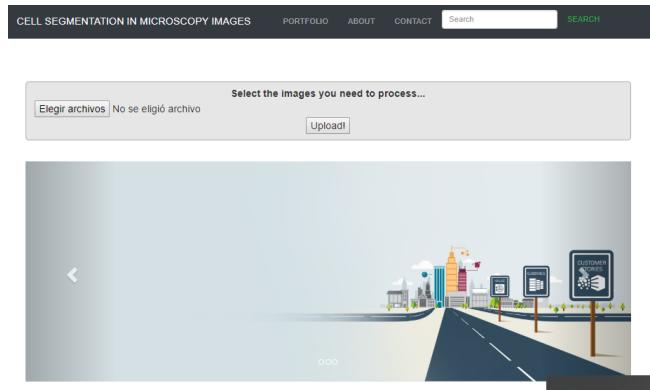


Fig. 5: Selector de imágenes y carrusel en página web

Images Gallery

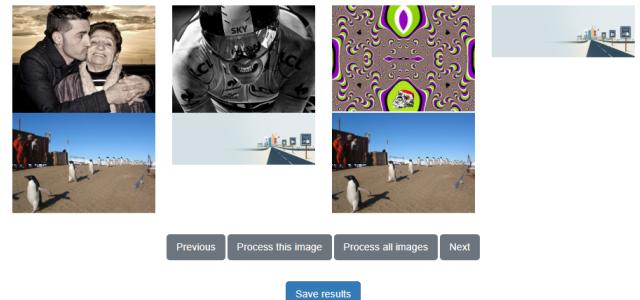


Fig. 6: Galería con imágenes seleccionadas

V. PROBLEMAS ENCONTRADOS

Para poder utilizar keras dentro del proyecto es necesario tener instalado previamente el paquete TensorFlow, pues en caso contrario no se podrán utilizar las funciones de keras. No obstante, TensorFlow no tiene la capacidad de adaptarse a todas las versiones de Python (de hecho solo lo hace con dos versiones específicas). Además de ello, TensorFlow está diseñada por defecto para arquitectura de 64 bits, mientras que Python por defecto viene preparado para arquitectura de 32 bits. En otras palabras, a la hora de descargar Python de la página oficial no existe la opción de elegir si se quiere el de 32 o el de 64 bits. Esta incompatibilidad genera problemas en la instalación. Por tanto, para poder resolver este problema existían dos caminos: Instalar algunas de las dos versiones de Python soportadas por TensorFlow y específicas para arquitectura de 64 bits. En cuyo caso era necesario desinstalar otras versiones de Python instaladas previamente en el equipo. Instalar el software Anaconda Navigator, el cual ayuda a resolver el problema de incompatibilidad de arquitecturas. En cuyo caso se debe recurrir a software de terceros y que en realidad no será utilizado para más que dicha instalación.

Por otra parte, el desconocimiento con que cuenta el equipo de trabajo en el aspecto de desarrollo web provoca incertidumbre a la hora de tomar decisiones. En el caso específico del framework necesario para desarrollo web en Python, se tienen bastantes opciones, pero Flask y Django cuentan con mayor popularidad. Flask tiene las características de ser sencillo de aprender, pero es un poco limitado en cuanto a funciones respecto a Django, que es más complejo de aprender, pero más completo en funciones. Dada esta dicotomía, se optó por elegir el más sencillo, pues para este contexto en específico se pueden lograr los objetivos con el mismo. No obstante, no sabemos si a futuro requeriremos de funciones un tanto más avanzadas y que no sean soportadas por Flask.

En la parte de carga de modelos con Keras, existen maneras distintas de guardar el modelo, por ello se debe tener cuidado del método que se escoga. Keras permite guardar únicamente los pesos del modelo, o bien, todo el modelo completo: sus pesos, entrenamiento, pruebas, etc. A la hora de cargarse se debe asegurar que se haya guardado de la segunda manera y no de la primera, ya que sino será imposible cargar un modelo.

En cuanto a OpenCV para la carga de imágenes no existe ningún problema, OpenCV está optimizado para cargar la gran variedad de imágenes. Con respecto a Pandas para la carga de CSV tampoco existe alguna limitación, de lo contrario, Pandas permite cargar muchos tipos distintos de CSV.

Por otra parte, a nivel de front-end se tomó la decisión de usar Bootstrap, el cual es un framework que facilita la generación de componentes más efectivos y visualmente más atractivos. Sin embargo, algunos de los componentes implementados requieren de librerías pre-compiladas de

JavaScript, CSS o JQuery. Esto condujo a problemas de versiones en el uso de dichas librerías, pues algunos de los componentes no eran compatibles con las versiones que usan otros componentes. Dada esta situación, algunos componentes debieron ser sustituidos por otros, buscando que las versiones se ajustasen. Aún realizando estas pruebas no se logró una integración total, por lo que el sistema hace uso de varias de estas librerías.

VI. CONCLUSIONES

Después de analizar los resultados de las pruebas realizadas podemos concluir que efectivamente realizar una prueba de concepto, previo a adentrarse en el desarrollo completo del proyecto, puede facilitar en gran medida el camino futuro. Esto pues, si con implementaciones sencillas, algunas de las herramientas no favorecen el desarrollo, mucho menos lo harán con implementaciones que sean más complejas.

También pudimos concluir que a la hora de realizar una integración entre back-end y front-end, el framework que mejor se adaptó a las necesidades fue Flask, aún cuando se supone que Django es más completo.

Por otra parte, se logró comprobar que el uso de keras no es tan complicado como se esperaba, al menos en este primer alcance. No obstante, se debe tener bastante cuidado en la instalación del mismo, para que luego no tenga problemas de ningún tipo.

Respecto al uso de la metodología Scrum, es muy recomendable que existan herramientas de uso libre que permitan tener tales funcionalidades, pues a pesar de no ser de pago, cuenta con las funciones necesarias para gestionar un proyecto de la mejor manera bajo esta metodología.

VII. ANEXOS

A. Enlace a GitHub

<https://github.com/saj11/CelularDivision>

B. Enlace a Zoho

<https://projects.zoho.com/portal/qaprojecttec/newlogin.dotodomilestones/132294900000023007/customview/6>

C. HTML generado con Doxygen

Debido a que la herramienta se ejecuta localmente, esta sección será mostrada presencialmente en la revisión.

VIII. BIBLIOGRAFÍA

REFERENCES

- [1] [HTTPS://GETBOOTSTRAP.COM/](https://getbootstrap.com/)
- [2] [HTTP://FLASK.POCO.Org/](http://FLASK.POCO.Org/)
- [3] [HTTPS://OPENCV.ORG/](https://OPENCV.ORG/)
- [4] [HTTPS://WWW.TENSORFLOW.ORG/INSTALL/](https://WWW.TENSORFLOW.ORG/INSTALL/)
- [5] <https://keras.io>