Diseño, desarrollo e implementación de una prueba de concepto para el despliegue de modelos analíticos en la nube publica de Scotiabank – Cloud Development Platform

Arcangel Marin Palacios

Informe de práctica para optar al título de Ingeniero de Sistemas

Asesor
Jeysson Pérez Gómez, Especialista en Gerencia Integral

Universidad de Antioquia
Facultad de Ingeniería
Ingeniería de Sistema
Medellín, Antioquia, Colombia
2022

Dedicatoria

Este logro es dedicado a aquellas personas que siempre han creído en mí, me han apoyado y motivado a ser cada día una mejor persona. Mis padres José(Q.E.P.D) y María, mis hermanos Diana y Abelardo, mi esposa Linda Zapata. Este es el resultado de muchos sacrificios en conjunto, por esta razón este logro no es solo mío, también les pertenece a ellos.

Agradecimientos

A mi madre María Palacios por sus sacrificios para sacarnos a delante, por su confianza, apoyo y sobre todo por su amor incondicional.

A mis familiares, mis hermanos Diana y Abelardo por apoyarme en los momentos fundamentales de mi crecimiento personal y profesional, por el ejemplo que me dieron desde niño.

A Linda Zapata por ser esa persona incondicional, por estar en los momentos más difíciles y felices durante muchos años, por su amor, por esos sacrificios y luchas que ha vivido a mi lado.

A Scotiabank por abrirme las puertas para poder desarrollar mis capacidades no solo en la ejecución de las practicas académicas si no en otros proyectos en los que he podido crecer profesional y personalmente.

A todo el equipo y compañeros de IT Analytics que me han guiado y apoyado para sacar adelante este reto. Especial agradecimiento a Edgar Lozano por su confianza, por sus consejos y asesoría.

A la universidad de Antioquia, docentes y compañeros de clase que han contribuido con mi formación académica y personal.

TABLA DE CONTENIDO

LISTA DE FIGURAS

SIGLAS, ACRÓNIMOS Y ABREVIATURAS

POC	Prueba de Concepto
CDP	Cloud Development Platform
IT	Información y Tecnología
ML	Machine Learning
SaaS	Software as a servicie
PaaS	Platform as a service
IaaS	Infrastructure as a service
UdeA	Universidad de Antioquia

RESUMEN

La computación en la nube permite utilizar redes de computadores remotos interconectados y alojados en Internet, que permiten almacenar procesar y administrar datos. Estos sistemas evitan invertir demasiados recursos y soporte de infraestructura física en la mayoría de los casos. Para el proyecto Pricing Customer View de Scotiabank el cual busca desarrollar una aplicación que permita asignar tasas personalizada de diferentes productos a los clientes, es importante explorar diferentes alternativas para desplegar modelos analíticos como servicios; una de esas alternativas es usar la nube pública del banco Cloud Development Platform (CDP) como plataforma para construir, integrar y desplegar dichos modelos. Este trabajo busca explorar CDP a través de una prueba de concepto (POC) que se desarrollará mediante diferentes actividades de investigación de información y desarrollo de software en diferentes componentes que se podrían utilizar como base para la solución del proyecto, una vez se logre evidenciar la viabilidad de usar esta plataforma para tal fin. Dichas actividades serán administradas bajo el marco de trabajo SCRUM como metodología en el proceso de construcción de la aplicación, obteniendo como producto final un microservicio desarrollado en Python desplegado en CDP, que permita ser consumido vía REST por otros microservicios y que contenga en su estructura un modelo de ML de prueba; demostrando la viabilidad de usar esta plataforma en la solución de la aplicación.

Palabras clave — **Computación en la nube, Microservicios, Modelos, prueba de concepto**

ABSTRACT

The Cloud computing allows the use of remote computer networks interconnected and hosted on the Internet, which allow data to be stored, processed and managed. These systems avoid investing too many resources and physical infrastructure support in most cases. For Scotiabank's Pricing Customer View project, which seeks to develop an application that allows customers to assign personalized rates for different products, it is important to explore different alternatives to deploy analytical models as services; one of those alternatives is to use the bank's public cloud Cloud Development Platform (CDP) as a platform to build, integrate and deploy such models. This work seeks to explore CDP through a proof of concept (POC) that will be developed through different information research activities and software development in different components that could be used as a basis for the project solution once the feasibility of using the platform for this purpose is demonstrated. These activities will be managed under the SCRUM framework as a methodology in the application construction process, obtaining as a final product a microservice developed in Python deployed in CDP, which allows it to be consumed via REST by other microservices and which contains a test ML model in its structure; demonstrating the feasibility of using this platform in the application solution.

Keywords — **Cloud Computing, Micro-services, Models, Proof of concept.**

I. INTRODUCCIÓN

Pricing Customer View es un proyecto de Scotiabank Colpatria que, desde el área de analítica y tecnología, busca desarrollar una aplicación que permita calcular y ofrecer a sus clientes una tasa optima a productos como lo son créditos de diferentes tipos (tarjetas, compra de cartera, rotativos, etc.). Esto mediante modelos analíticos que tendrán en cuenta una serie de variables tanto de mercado como del cliente; modelos que serán desplegados a través de microservicios y que podrán ser consumidos por diferentes aplicaciones internas del banco, las cuales se encargan del onboarding del cliente a dichos productos. Con el fin de ir definiendo una arquitectura y los sistemas u componentes que permitan llevar a cabo dicha solución, se han hecho una serie de pruebas de concepto a través de plataformas de terceros que permiten el despliegue y administración de modelos analíticos y de machine learning (ML). Pruebas que han presentado algunas dificultades técnicas ya que la infraestructura on-premise administrada por los equipos de tecnología debe ser soportada y customizada para tal fin, lo cual no ha permitido hasta cierto punto, fluidez en la ejecución de las pruebas, lo que traería consigo posibles retrasos en el road-map del proyecto.

Uno de los objetivos a nivel tecnología del banco es la modernización de las plataformas. Dentro de esta modernización está la de pasar de aplicaciones construidas a partir de programas monolíticos a programas con un estilo de arquitectura basado en microservicios, además, tener una visión Cloud First (primero en la nube) en la que por medio de sistemas de computación en la nube puedan ser desplegados los microservicios para las aplicaciones, permitiéndole a los equipos concentrarse en el desarrollo y entrega de las soluciones ágilmente, más que en el soporte de infraestructura.

La organización cuenta con una plataforma de nube publica (Cloud Development Platform) alojada en Microsoft Azure, pensada con esta finalidad, pero hasta el momento solo ha sido explorada en su gran mayoría para desplegar aplicaciones de tipo web basadas en JAVA y Node JS a través de pipelines automatizados de integración y despliegue continuo. Sin embargo, en las fuentes de información interna para los equipos de desarrollo en el área de tecnología, no existe suficiente documentación que indique el proceso para poder desplegar modelos analíticos y predictivos a

través de aplicaciones escritas en Python. Teniendo en cuenta que Python como lenguaje de programación, es uno de los que mejor se adaptaría a los equipos de científicos de datos para poder construir dichos modelos y poder desplegarlos a través de microservicios en la plataforma CDP. Dicho lo anterior surge la necesidad de explorar CDP con el fin estudiar la viabilidad de desplegar en ella servicios en Python que permitan integrar modelos analíticos y de machine learning al pipeline de DC/IC, el cual ya es usado por otro tipo de aplicaciones. Brindando así, otra vía como alternativa de solución al problema inicial del proyecto que busca calcular la tasa optima de los productos de crédito a los clientes.

De esta manera, en este trabajo se presentará el desarrollo de una prueba de concepto que permita explorar la plataforma CDP como una alternativa para desplegar microservicios que integren modelos analíticos o de ML en su lógica. Esto por medio de una consulta inicial en la documentación de los diferentes servicios y configuraciones que presta la plataforma y la puesta en marcha de desarrollos prácticos en esta, que permitan interactuar en caliente con los diferentes componentes hasta lograr crear una base para el flujo de datos que podría llegar a tener la aplicación. Actividades que serán desarrolladas bajo el marco de trabajo SCRUM durante un tiempo no mayor a 6 meses.

II. OBJETIVOS

A. Objetivo general

Diseñar, desarrollar e implementar una prueba de concepto que permita explorar la nube pública del banco Scotiabank CDP como plataforma para el despliegue de modelos analíticos a través de microservicios, por medio de las herramientas de integración, entrega y despliegue continuo vinculadas a CDP.

B. Objetivos específicos

- Explorar las herramientas y servicios disponibles en la nube pública del banco Cloud Development Platform para el despliegue de microservicios.
- Diseñar e identificar una arquitectura base que garantice el flujo de datos en la prueba de concepto.
- Desarrollar e implementar microservicios que permitan orquestar el flujo de datos en la prueba de concepto según la arquitectura base definida.
- Desplegar en CDP cada uno de los componentes creados a través del pipeline de Integración y despliegue continuo.

III. MARCO TEÓRICO

Un modelo de machine learning, es el producto obtenido después de entrenar un algoritmo computacional a partir de datos. Dicho modelo espera unos inputs o entradas y devuelve una salida de acuerdo con el entrenamiento realizado [1]. Muchas empresas han empezado a adoptar este tipo de tecnologías para optimizar sus procesos, sin embargo, no es una tarea sencilla, sobre todo el llevar estos modelos a un ambiente productivo. Uno de los pasos más importante es el despliegue, este es necesario para hacer un uso eficaz de los modelos de ML en producción. Estudios demuestran que gran parte de los proyectos de ML fallan, al no tener éxito en sus despliegues [2]. Cada día los equipos de desarrollo buscan nuevas alternativas para lograr eficiente mente este proceso, una de estas alternativas es la computación en la nube.

La computación en la nube es utilizada para almacenar, manejar y procesar datos haciendo uso de redes de computadoras remotas, a las que generalmente se puede acceder a través de Internet. Los usuarios de este tipo de tecnologías pueden escoger entre configuraciones de nube privada, pública o hibrida según sus requisitos [3]. Dentro de los proveedores de este tipo de servicios se encuentran compañías como Microsoft, Amazon, IBM, Salesforce, SAP, entre otros.

Existen varios segmentos de computación en la nube, entre los cuales están El software como servicio (SaaS), Plataforma como servicio (PaaS) e Infraestructura como servicio (IaaS).

El Software como servicio o en ingles Software as a servicie (SaaS), permite a los usuarios interactuar con aplicaciones basadas en la web a través de Internet sin necesidad de instalar aplicaciones adicionales en su computadora. Por otra parte, en las Plataformas como Servicio o en ingles Platform as a Service (PaaS) los usuarios no se preocupan por elementos como la infraestructura o los sistemas operativos, este es otro tipo de servicio en la nube pensada para los desarrolladores de aplicaciones, en donde tienen la posibilidad de desarrollar, testear, desplegar, alojar y mantener aplicaciones a través de plataformas en internet sin necesidad de instalar software en sus computadores [4]. El tercer segmento de computación en la nube es Infraestructura como Servicio o en ingles Infrastrcture as a Service (IaaS), en este, los vendors se encargan de la infraestructura y los usuarios de los sistemas operativos; la infraestructura como servicio es un tipo

de computación en la nube que ofrece recursos de proceso, almacenamiento y redes a petición, que son de pago por uso [5].

Las PaaS como por ejemplo Cloud Foundry (CF), la cual es una de estas plataformas como servicios de código abierto que suele ser utilizada por los equipos de desarrollo para llevar las aplicaciones escalables desde la construcción y despliegue hasta su ejecución [6]; Estas aplicaciones tienen diferentes estilos arquitectónicos, pero uno de los más usados en entornos de computación en la nube son los microservicios. Los microservicios son estilos de arquitectura que estructuran una aplicación como una colección de servicios, los cuales presentan las siguientes características:

- Altamente mantenibles y testeables.
- Desacoplados
- Independientemente desplegables
- Organizados en torno a las capacidades del negocio
- Administrados por equipos pequeños.

Los microservicios han ganado popularidad en muchos campos en el desarrollo de software, dado los retos que existen de construir aplicaciones y plataformas altamente escalables y complejas [7]. Estos microservicios permiten que los equipos de desarrollos puedan concentrarse en solucionar funcionalidades específicas de la aplicación, que combinada con metodologías agiles como SCRUM y paradigmas de desarrollo de software como DevOps hagan más eficiente y rápida la entrega de valor a los clientes.

DevOps es automatización de extremo a extremo (End-to-End) en el desarrollo y la entrega de software [8]. La adopción de este paradigma de producción de software ha brindado ventajas en la entrega rápida y continua de aplicaciones, sin perder los atributos de calidad requeridos por el cliente [9]. De este hacen parte los equipos de desarrollos, la operación enfocada en la administración de los lanzamientos, despliegues y los grupos responsables de probar la calidad del software. Todo enmarcado dentro de un ciclo que pasa por ciertas etapas a través de pipelines de integración, entrega y despliegue continuo que pasan por el desarrollo, construcción, integración, pruebas, lanzamientos, despliegue y operación del software.

Por otra parte, SCRUM es un marco de trabajo que proporciona herramientas mediante reglas y tareas durante las iteraciones que se dan en un proyecto, esto con el fin de asegurar su correcta ejecución. Este marco de trabajo se compone por los equipos SCRUM y sus roles, los eventos, artefactos y reglas asociadas [10]. Según la guía SCRUM este consta de los siguientes eventos:

- Sprint
- Sprint Planning
- Daily Scrum
- Sprint Review
- Sprint Retrospective

Los roles presentes en este marco de trabajo son Scrum Master, Product Owner y Development Team. Mientras que los artefactos están definidos por el Product Backlog, Sprint Backlog y el Incremento [11].

Una de las aplicaciones usadas para la gestión de proyectos mediante marcos de trabajo ágil es Jira de Atlassian, la cual impulsa la colaboración entre los miembros del equipo desde el diseño hasta la finalización. Permitiendo que los equipos puedan planificar, asignar, supervisar y gestionar el trabajo [12].

En la búsqueda de desplegar de manera ágil aplicaciones o software a través de los equipos de desarrollo en las diferentes áreas de tecnología, Scotiabank cuenta con CDP, la cual es la plataforma de computación en la nube publica del banco para la construcción, despliegue y ejecución de microservicios en la nube. Esto a través de pipeline de integración y despliegue continuo vinculados con CDP.

IV. METODOLOGÍA

Para el desarrollo de este proyecto, se siguió la metodología Scrum en un marco de trabajo que buscó definir para el equipo de IT Analytics un conjunto de buenas prácticas en un proceso de desarrollo de software iterativo (Ver figura 1). El proyecto Pricing Customer View se conformó por varios equipos, entre los cuales se encontraban el equipo de la *Fabrica Digital* encargados de la solución de cara al front-end, los cuales serían los consumidores de los servicios de pricing, el equipo de *Negocio y Datascientist* encargados de levantar los requerimientos o necesidades de lado del Negocio y el diseño, desarrollo e implementación de los modelos predictivos con las reglas de asignación de tasas. Finalmente el equipo de *IT Analytics* encargados de la infraestructura y desarrollo de la solución en la cual se pudieran desplegar modelos de ML como servicios y dentro del cual se centraron algunas de las tareas principales para dar solución al problema abordado desde el proyecto de prácticas.

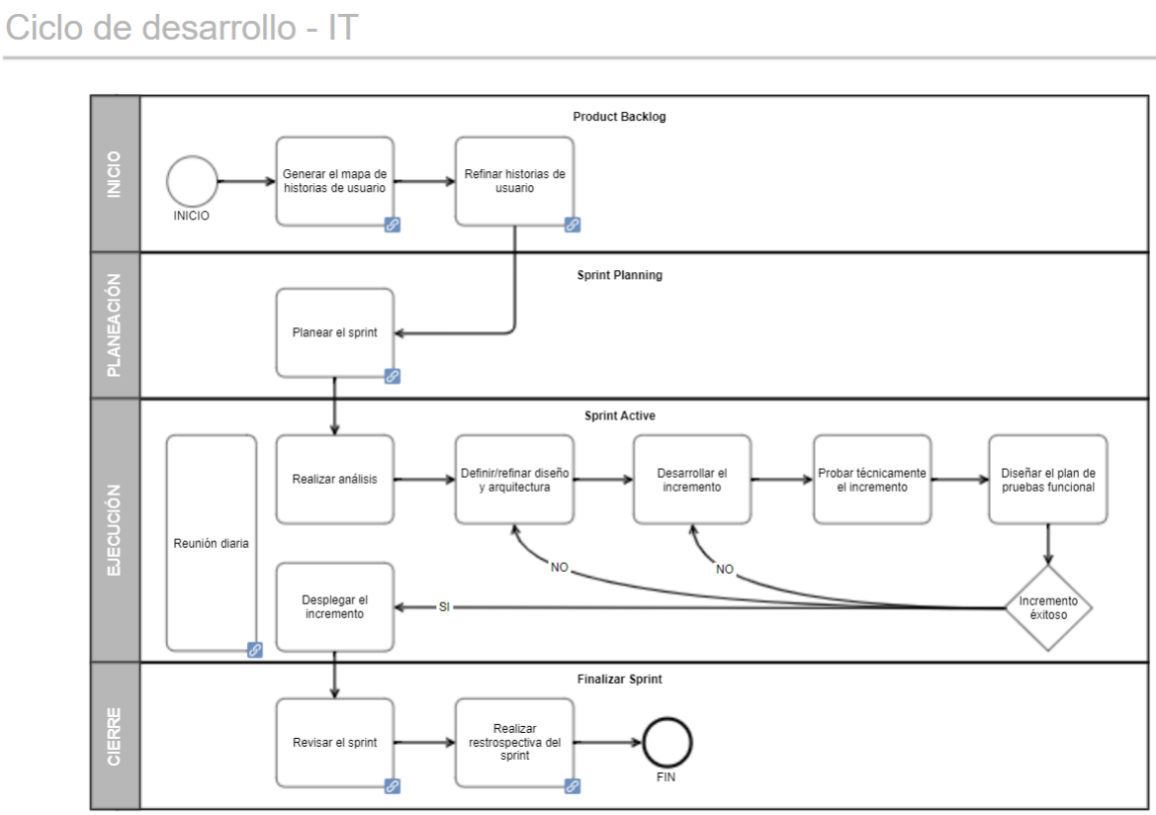

Fig. 1. Marco de trabajo equipo IT Analytics.

En la figura anterior se puede ver el flujo del marco de trabajo utilizado, el cual empezó definiendo unas historias de usuarios y tareas iniciales consignadas en el Product Backlog. Historias y tareas que garantizaron la culminación exitosa del proyecto. Esto en un proceso de planeación ejecutados dentro de los Sprint planning. Los Sprint los definimos de 15 días, dentro de los cuales se ejecutaron las tareas planeadas. Cada día se hacían seguimiento a las tareas del sprint activo validando el incremento que se hacía, hasta finalizar este con la revisión dentro de la ceremonia del sprint review.

Durante los primeros Sprint se ejecutaron tareas de investigación y revisión de información relacionada con el despliegue de microservicios en Python en la plataforma de CDP, ya que para el despliegue de microservicios en Java ya el equipo contaba con la experiencia. la información más relevante se iba recolectando y socializando con el equipo. Mientras tanto otros miembros del equipo de desarrollo seguían con el plan inicial en paralelo que buscaba usar una plataforma de contratada externa para la administración y despliegue de modelos de ML; con la investigación definida en las tareas se buscaba ver la viabilidad de hacer la implementación en un entorno de desarrollo y despliegue propio(CDP).

Después de tener la suficiente información, se diseñó una arquitectura inicial en la que se pudiera hacer una prueba de concepto que permitiera ver la viabilidad de usar CDP. Dicha arquitectura permitió identificar los microservicios que se iban a ir desarrollando en los Sprints siguientes, dando foco en el microservicio construido en Python y que contendría el modelo de ML. Posterior a esto empezaron las tareas de desarrollo y pruebas en el pipeline de IC/DC; las herramientas de desarrollo que se utilizaron fueron el Visual Studio Code con algunos plugins de Python para implementar el código del microservicio en Python, también se utilizó la herramienta Intellij Idea necesario para los desarrollos de los microservicios en Java con Sprint Boot. El versionado del código para los desarrollos del proyecto se hicieron con Bitbucket el cual está integrado con el pipeline. El seguimiento de las historias y las tareas se realizaron dentro de la plataforma de Jira de Atlassian, la cual permitía llevar el control del progreso de estas durante la ejecución del proyecto.

Dentro de CDP se tenía la herramienta Accelerator la cual permitía hacer la configuración del proceso de construcción de los artefactos que se iban creando al igual que el despliegue e

integración continua. Constantemente se tuvo que estar probando cada desarrollo del código para validar que las pruebas en el pipeline se ejecutaran de manera correcta, lo cual permitía ver si sí era posible el despliegue de los servicios con los modelos de ML en el pipeline.

En el proceso de desarrollo se iba versionando el código en ramas de tipo ¨feature¨, una vez se tenía algo estable a través de Accelerator se ejecutaba el pipeline vinculándolo a la rama en la cual se quería probar. Una ves se desplegaba el microservicio se hacía pull request desde las ramas ¨feature¨ a la rama ¨develop¨ en la cual se iban pasando los cambios estables de la aplicación. La solución propuesta en la prueba de concepto se iba probando por medio de Postman la cual permitía hacer peticiones http Rest a los servicios.

V. RESULTADOS Y ANÁLISIS

Como resultados de este proyecto de prácticas se logró que el equipo de IT Analytics del proyecto de Pricing Customer View encontrara una alternativa que permitiera el despliegue de modelos de ML a través de microservicios. Los resultados se dieron después de una serie de iteraciones dentro de 10 sprints de trabajo aproximadamente y que pasaron por las siguientes etapas:

Etapa de Investigación: En esta fase se logró consultar información en el repositorio de documentación interna, acerca de las características de la plataforma de CDP y las diferentes herramientas con las que esta cuenta. Se pudo identificar que era posible desplegar además de Java y Node js, aplicaciones de Python a partir de los templates de aplicaciones a través de Accelerator (ver figura 2).

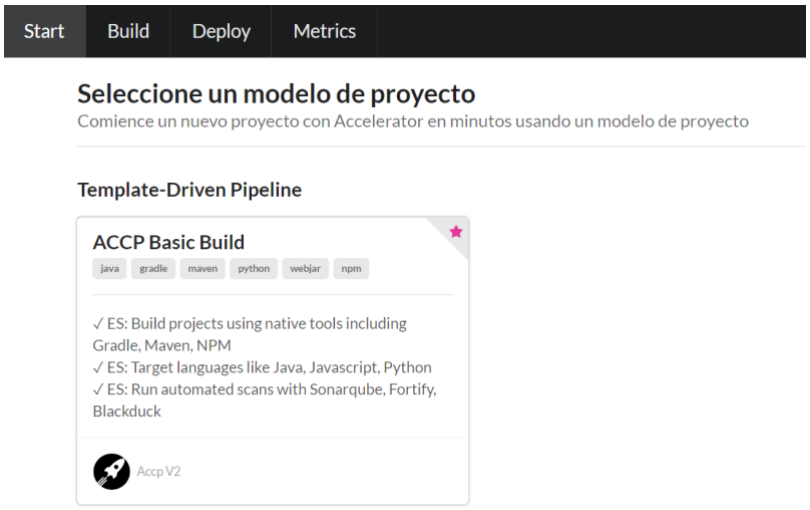

Fig. 2. Vista de inicialización de apps a partir de templates en Accelerator.

Encontramos que Accelerator nos permitiría implementar la metodología Devops trabajando en el pipeline, el ciclo de desarrollo de software desde la estructura inicial de la aplicación (Start), la construcción y versionamiento de los artefactos (Build), Despliegue de la aplicación (Deploy) y monitoreo de la aplicación por el equipo operativo una vez la aplicación esté en funcionamiento (Metrics). Dentro de las principales características se encontró la autogestión de los diferentes recursos y servicios inegrados a la plataforma, como por ejemplo el auto escalado de las aplicaciones, la administración de los conteiners donde se despliegan estas, el auto monitoreo de

la salud de los microservicios o health check, la administración del rendimiento e integración con diferentes herramientas como logging, servicios de caché, de almacenamiento, entre otros.

Etapa de Diseño: Una vez identificadas las características de la plataforma y algunas consideraciones para el despliegue de aplicaciones tipo Python, se procedió con el diseño de una arquitectura base que permitiera hacer la prueba de concepto con el fin de identificar componentes básicos que intervendrían en el flujo de datos de la solución y las configuraciones adicionales en la comunicación de los diferentes ambientes tanto de los componentes internos como externos a la aplicación.

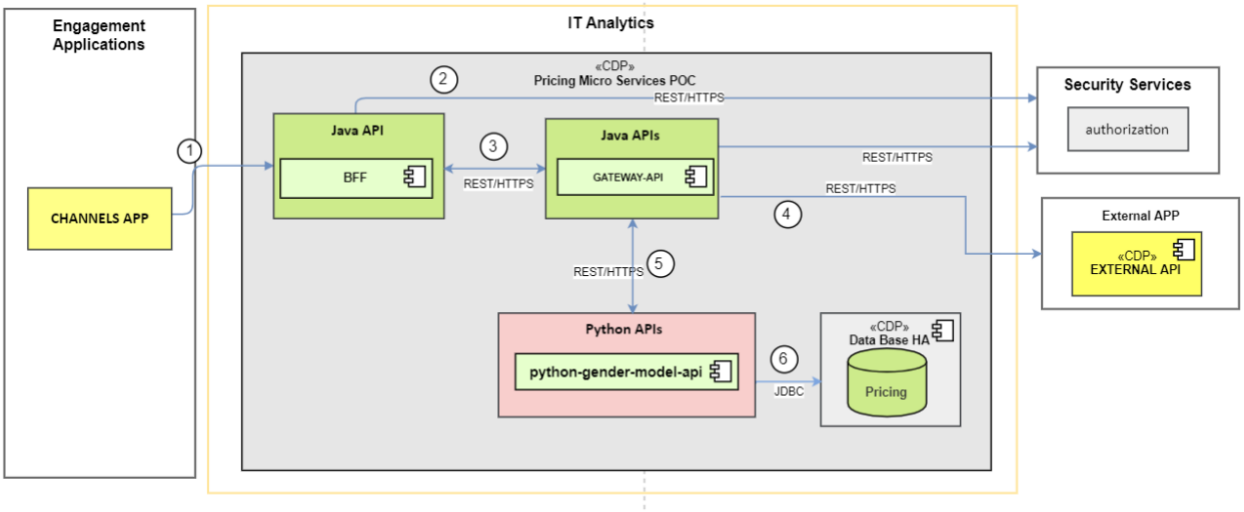

Fig. 3. Arquitectura prueba de concepto.

En la arquitectura se puede observar que la línea naranjada encierra la zona de interacción del equipo de IT Analytics (ver figura 3), donde los componentes de color verde fueron desarrollados y desplegados en el ambiente de CDP para validar inicialmente la comunicación interna y la comunicación con algunas aplicaciones externas que interactuarían en la solución final. Gran parte del foco de esta proyecto de practica recayó sobre el componente de color rojo, en el cual se encontraba el microservicio *python-gender-model-api* el cual era una aplicación de tipo Python y que albergaba el modelo de ML de prueba.

Etapa de desarrollo: Con el conocimiento base del funcionamiento de la plataforma de CDP e identificado los componentes que interactuarían dentro de la prueba de concepto se procedió con

las tareas de desarrollos a partir del 4 sprint. Según la arquitectura se debía desarrollar un microservicio (*BFF-API*) que representara el punto de contacto desde las aplicaciones externas para consultar el precio óptimo de un producto especifico. En este caso y para efectos de la POC este servicio solo tenía que recibir vía REST método POST la variable que contuviera el nombre de una persona, ya que el modelo de prueba había sido entrenado para predecir el género de una persona (Masculino, Femenino) a partir de su nombre, por ejemplo si como entrada se le pasaba el nombre ¨María¨ este devolvía como salida ¨Femenino¨, en el caso de que se le pasara como entrada el nombre ¨Mario¨ este devolvía como salida ¨Masculino¨. Este microservicio a su vez debía redirigir la petición inicial a un microservicio intermedio (*GATEWAY-API*) Encargado a su vez de simular peticiones a aplicaciones externas y de redirigir la petición enviada desde el BFF-API al microservicio desarrollado en Python y que albergaba el modelo. Los anteriores servicios fueron desarrollados en Java con SprintBoot y no presentaban mayor dificultad puesto que ya todo el equipo estaba acostumbrado a trabajar esta tecnología en la infraestructura de CDP.

Finalmente el microservicio *python-gender-model-api* el cual representaba el reto mayor, requirió identificar los archivos de configuración para poder integrarlo en el pipeline, de tal manera que se ejecutaran por ejemplo los escáneres de seguridad y calidad del código, análisis de test unitarios, y configuraciones que dieran las características a los ambientes en los contenedores donde estos se desplegaban, además de definir la estructura que permitiera la administración e identificación de los modelos dentro de los archivos y carpetas que hacían parte de la aplicación.
El primer paso dentro del desarrollo era crear el proyecto a partir del template especifico del pipeline de Accelerator(Ver figura 4)

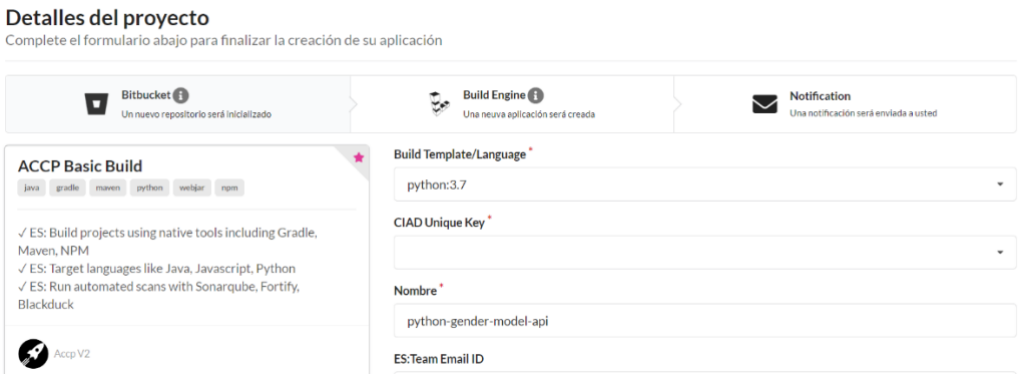

Fig. 4. Inicialización de aplicación en pipeline de Accelerator.

Al inicializar este proceso se crearon las configuraciones iniciales del pipeline y se creó automáticamente el repositorio en Bitbucket asociado al identificador del proyecto. Desde allí empezó el proceso de desarrollo, versionado, ejecución del pipeline y pruebas de funcionamiento de la aplicación en un flujo iterativo durante los diferentes sprint.

Estructura del microservicio *python-gender-model-api*: Este microservicio fue desarrollado con la librería FastApi para permitir recibir peticiones REST y facilitar la orquestación del flujo hasta el consumo del modelo de prueba y comunicación con la base de datos de prueba igualmente. La librería usada para permitir servir la aplicación fue uvicorn.

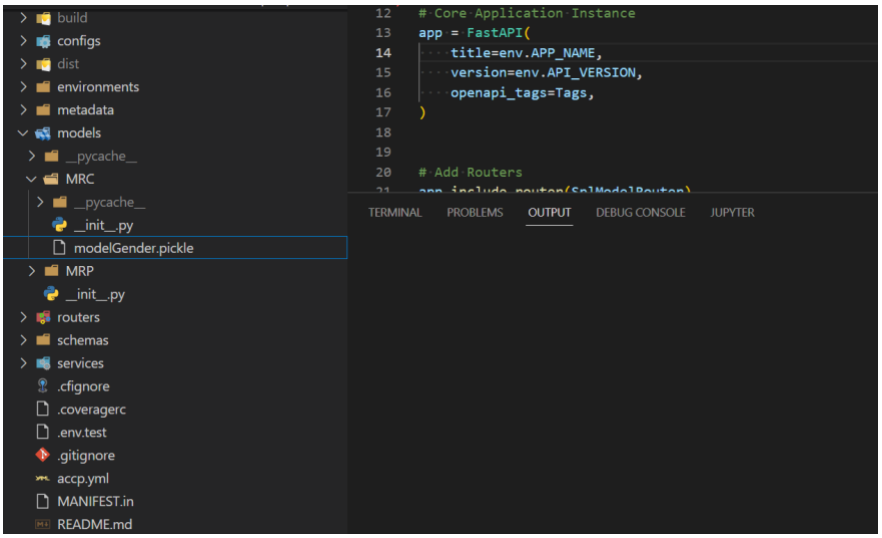

Fig. 5. Estructura microservicio python-gender-model-api.

Como se puede observar en la Figura 4 se tuvieron que crear algunas carpetas y archivos de configuración para que pudiera desplegarse de manera exitosa el microservicio en el pipeline. Configuraciones desde el paso de creación de artefactos a través de la etapa de *Build* Como se ve en la figura 5 y en la cual se hace el proceso de validaciones a nivel de scanners de seguridad y calidad.

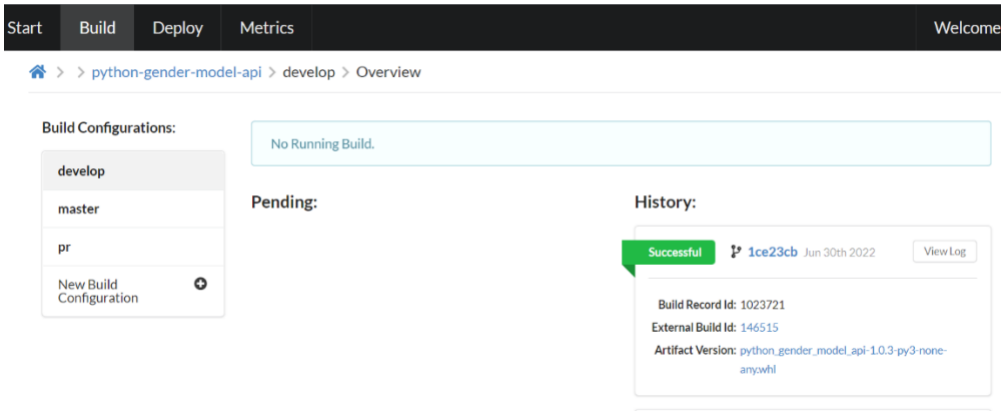

Fig. 6. Construcción de artefacto exitoso en pipeline de Accelerator.

Y además configuraciones para el paso de despliegue (ver imagen 6) y en el cual se especifican las características a nivel de infraestructura, rendimiento, almacenamiento, proceso de auto escalamiento, entre otros.

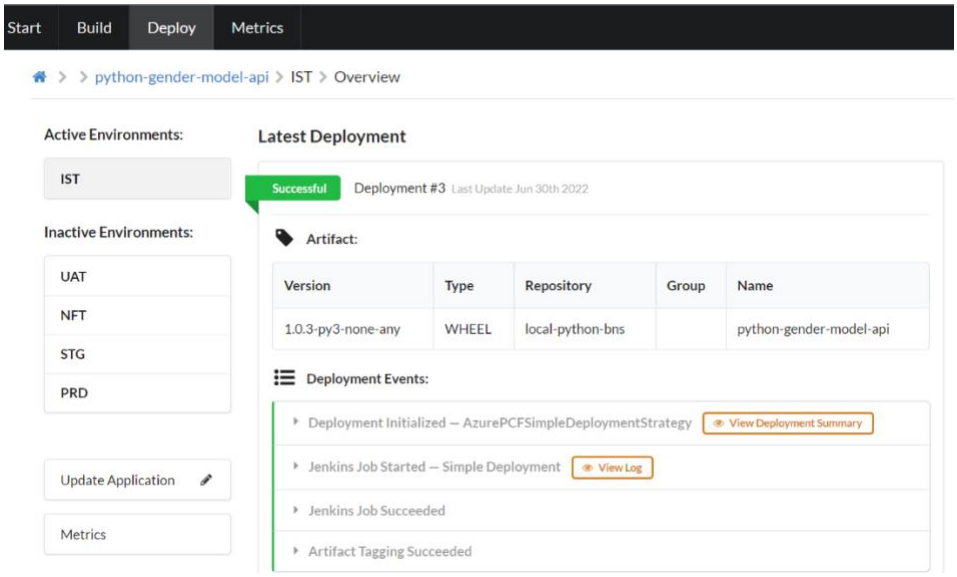

Fig. 7. Despliegue de artefacto exitoso en pipeline de Accelerator.

Finalmente a través de la consola de administración de la aplicación una vez pasó por todo el flujo del pipeline, se podía configurar el auto gestionado para los eventos de escalado, asignación de memoria , pausa, reinicio y apagado de la aplicación (ver figura 7), características de las bondades de la plataforma de CDP.

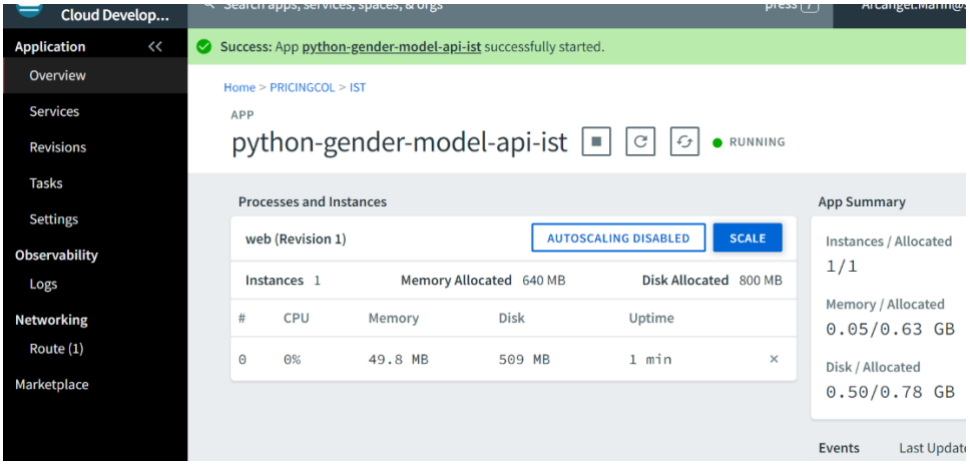

Fig. 8. Administración microservicio python-gender-model-api desde CDP.

Para finalizar se ejecutaron las pruebas desde la herramienta de Postman haciendo peticiones a la url generada para el servicio, enviándole como parámetro la variable nombre y obteniendo como respuesta la predicción hecha por el modelo de prueba entrenado para tal fin, como se mencionó anteriormente (ver figura 8).

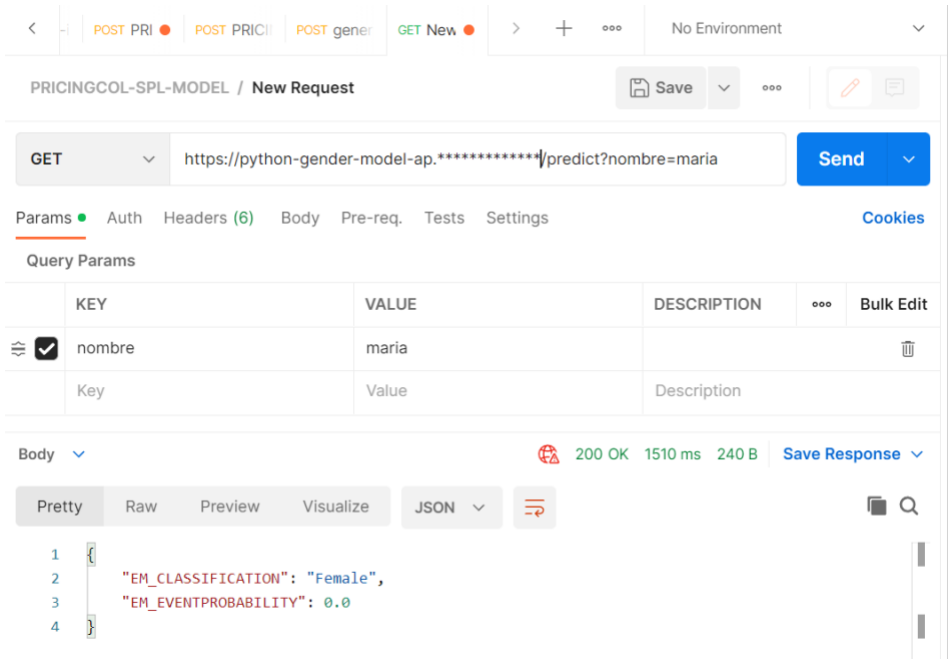

Fig. 9. Prueba de petición a microservicio python-gender-model-api.

Terminada la parte más importante de esta prueba de concepto la cual consistía de validar si era posible desplegar modelos de ML en microservicios Python desplegados en CDP, se integraron todos los microservicios desarrollados para tal fin, mostrando el funcionamiento del flujo de datos propuesto en la arquitectura de la POC.

Como se pudo demostrar en los resultados obtenidos en el desarrollo de los sprint planeados, dejaron una segunda alternativa como propuesta para desplegar modelos de ML como servicio. Alternativa que no había surgido antes de las dificultades presentadas en la primera fase de exploración en otras plataformas de terceros y que gracias a este proyecto de prácticas pudo materializarse generando un gran valor para el equipo de IT Analytics y el proyecto de Pricing Customer View en general.

VI. CONCLUSIONES

Con la finalización de este proyecto se pudo obtener para la iniciativa de Pricing Customer View una alternativa viable que garantice el despliegue de modelos de ML como servicios en la plataforma de Cloud Computing disponible para el banco CDP, dentro de un ecosistemas de tecnologías que garantiza la alta disponibilidad, mantenibilidad y gestión de las aplicaciones. Esto, gracias a la prueba de concepto ejecutada, probando el flujo de datos en la arquitectura inicial planteada.

La investigación, exploración y documentación inicial de herramientas y servicios disponibles en CDP para el despliegue de aplicaciones Python fue de gran valor parea el equipo de IT Analytics y en general para la gerencia de tecnología, ya que abre un camino importante para la implementación de futuras aplicaciones con necesidad de incorporar tecnologías disruptivas que satisfagan las necesidades actuales en materia de implementaciones basadas en datos y machine learning.

El uso de metodologías agiles como Scrum y Devops permiten que el desarrollo de software sea más eficiente y que los equipos de trabajo se enfoquen en implementar soluciones que aporten gran valor a las entidades. Esto se evidencio a la hora de ejecutar la prueba de concepto bajo el marco de trabajo definido y el uso del pipeline de integración y despliegue continua de Accelerator que permitió observar cómo fácilmente se llevaba una aplicación desde el desarrollo hasta su despliegue en un ambiente operativo.

Si bien la prueba de concepto deja fuertes bases para continuar con la alternativa desarrollada, puede que sea necesario perfeccionar la estructura del microservicio en Python con el fin de que sea más robusta y garantice todos los estándares de calidad y seguridad exigidos por el banco, de tal manera que pueda pasar en un futuro todas las pruebas que normalmente se suelen hacer en la gerencia de tecnología antes de lanzar una aplicación en el ambiente productivo.

REFERENCIAS

[1] IBM, Que es machine learning?, https://ibm.co/3BSsu8d (accessed Sep. 15, 2022)

[2] Heymann, H., Kies, A. D., Frye, M., Schmitt, R. H., & Boza, A. (2022). Guideline for Deployment of Machine Learning Models for Predictive Quality in Production. *Procedia CIRP*, *107*, 815–820. https://doi.org/https://doi.org/10.1016/j.procir.2022.05.068

[3] Sukesh, M., Naveen Kumar, R., & Lokanatha Reddy, C. (2021). Development and deployment of real-time cloud applications on red hat OpenShift and IBM bluemix. Materials Today: Proceedings. https://doi.org/https://doi.org/10.1016/j.matpr.2021.01.458.

[4] Modisane, P., & Jokonya, O. (2021). Evaluating the benefits of Cloud Computing in Small, Medium and Micro-sized Enterprises (SMMEs). Procedia Computer Science, 181, 784–792. https://doi.org/https://doi.org/10.1016/j.procs.2021.01.231.

[5] Microsoft Azure, ¨*¿Qué es IaaS?*¨. https://azure.microsoft.com/es-es/overview/what-is-iaas/#overview (accessed Jun. 25, 2022)

[6] Cloud Foundry, ¨Cloud Foundry Overview¨, https://docs.cloudfoundry.org/concepts/overview.html (accessed Jul. 02, 2022)

[7] Nasser, A. A., Rashad, M. Z., & Hussein, S. E. (2020). A Two-Layer Water Demand Prediction System in Urban Areas Based on Micro-Services and LSTM Neural Networks. *IEEE Access*, *8*, 147647–147661. https://doi.org/10.1109/ACCESS.2020.3015655.

[8] Ebert, C., Gallardo, G., Hernantes, J., & Serrano, N. (2016). DevOps. *IEEE Software*, *33*(3), 94–100. https://doi.org/10.1109/MS.2016.68.

[9] Altunel, H., & Say, B. (2021). Software Product System Model: A Customer-Value Oriented, Adaptable, DevOps-Based Product Model. *SN Computer Science*, *3*(1), 38. https://doi.org/10.1007/s42979-021-00899-9

[10] Gaete, José, Villarroe, Rodolfo, Figueroa, Ismael, Cornide-Reyes, Héctor, & Muñoz, Roberto. (2021). Enfoque de aplicación ágil con Serum, Lean y Kanban. *Ingeniare. Revista chilena de ingeniería*, *29*(1), 141-157. https://dx.doi.org/10.4067/S0718-33052021000100141.

[11] K. Schwaber and J. Sutherland. La guía de *Scrum* , https://bit.ly/3LlntrX (accessed Jul. 02, 2022).

[12] Atlasssian, Una breve presentacion de Jira, https://bit.ly/3Bf5bUC (accessed Jul. 03, 2022).