



## **Pruebas automatizadas a aplicativo CRM**

Juan Alejandro Ríos Pérez

Trabajo de prácticas presentado como requisito parcial para optar al título de:  
**Ingeniero de Sistemas**

Asesor

Gustavo Andrés Marín Lopera, Ingeniero de Sistemas

Universidad de Antioquia  
Facultad de Ingeniería  
Ingeniería de Sistemas UdeA  
Medellín  
2023



## Referencia

- [1] J. A. Ríos Pérez, "Pruebas automatizadas a aplicativo CRM", Presencial, Ingeniería de Sistemas, Universidad de Antioquia, Medellín, 2023.

Estilo IEEE (2020)



**Repositorio Institucional:** <http://bibliotecadigital.udea.edu.co>

Universidad de Antioquia - [www.udea.edu.co](http://www.udea.edu.co)

**Rector:** John Jairo Arboleda Céspedes.

**Decano/Director:** Julio César Saldarriaga.

**Jefe departamento:** Diego Jose Luis Botia Valderrama.

El contenido de esta obra corresponde al derecho de expresión de los autores y no compromete el pensamiento institucional de la Universidad de Antioquia ni desata su responsabilidad frente a terceros. Los autores asumen la responsabilidad por los derechos de autor y conexos.

## **Agradecimientos**

Gracias a la vida y a lo que me haya permitido vivir esta maravillosa experiencia, a mis padres Leydy y Juan, a mi hermana Angie quienes me han acompañado durante este recorrido y amo profundamente, al igual que al resto de mis familiares y amigos. Agradecer a mi alma máter, para mi es un honor pertenecer y contribuirle un poco, aunque jamás tendré como retribuir todo lo que me ha dado. Agradezco también al Colegio Parroquial Emaús ya que en parte debido a lo allí aprendido y vivido logré ingresar a la universidad. Doy gracias a Colombia porque sin la contribución de todos a los diferentes programas sociales y a la universidad pública difícilmente podría haber aspirado a ser profesional. Muchas gracias a todas las personas, lugares y momentos que directa o indirectamente me han permitido optar por cumplir este objetivo. Gracias al profe Gustavo por ser mi asesor y apoyo en este y otros proyectos que hemos construido juntos. Gracias a TCS, a Catalina quién me brindó mi primer empleo, me formó en automatización de pruebas y en muchos conocimientos adquiridos que apliqué en este trabajo. Gracias a la compañía Choucair por darme la oportunidad de realizar mis prácticas profesionales allí, por todo los conocimientos que me han aportado y lo feliz que he sido trabajando para ustedes, a Norma por en nombre de la compañía hacerlo posible, a Katerinne y Luisa por apoyarme en los diferentes procesos como asesoras. No caben, ni existen palabras que logren describir lo agradecido que me encuentro con todos y cada uno de ustedes y lo afortunado que me siento de haberlos tenido en mi camino, espero y aspiro a que la vida me permita agradecerles de muchas más formas y que logre este y muchos más objetivos y sueños por los cuales seguir agradeciéndoles. Muchísimas gracias por tanto.

## TABLA DE CONTENIDO

RESUMEN	7
ABSTRACT	7
I. INTRODUCCIÓN	8
II. OBJETIVOS	9
III. MARCO TEÓRICO	10
IV. METODOLOGÍA	12
V. RESULTADOS	16
VI. CONCLUSIONES	19
REFERENCIAS	20

## LISTA DE FIGURAS

Fig. 1. Modelo de DevOps	10
Fig. 2. Ciclo de un SPRINT en scrum	12
Fig. 3. Actividades realizadas en la práctica académica bajo metodología Choucair	13
Fig. 4. Estructura de proyecto POM	16
Fig. 5. Ejemplo feature	17
Fig. 6. Ejemplo clase Page	17
Fig. 7. Ejemplo steps	18

## SIGLAS, ACRÓNIMOS Y ABREVIATURAS

<b>API</b>	Application Programming Interface
<b>CRM.</b>	Customer Relationship Management
<b>CI</b>	Continuous Integration
<b>CD</b>	Continuous Deployment
<b>POM</b>	Page Object Model
<b>IDE</b>	Integrated Development Environment
<b>PO</b>	Product Owner
<b>MVP</b>	Minimum Viable Product
<b>REST</b>	Representational State Transfer
<b>SOLID</b>	Single responsibility, Open-closed, Liskov substitution, Interface segregation and Dependency inversion
<b>UdeA</b>	Universidad de Antioquia

---

## RESUMEN

En este documento se presenta el trabajo realizado durante las prácticas académicas en las cuales se propuso el validar la calidad de un módulo de un CRM haciendo uso de herramientas de automatización de pruebas e integrando estas con el ciclo de vida del desarrollo del software, buscando mejorar la eficiencia de los procesos de pruebas conocidos también como certificación. Se explican los conceptos relacionados con este tema en los cuales entre otros se encuentran, DevOps, pipelines, CI/CD, Gherkin y de forma general cómo se implementaron las pruebas trabajando en una metodología ágil similar a scrum, con el framework Webdriver IO, bajo el patrón POM en TypeScript, utilizando un repositorio para el control de versiones, además de otros conocimientos adquiridos a raíz del trabajo como la automatización a servicios REST. Finalmente, se destaca que además de cumplir con los objetivos planteados al inicio de las prácticas y los conocimientos adquiridos y puestos en práctica en este trabajo, el llevarlo a cabo ha permitido apoyar procesos de adopción de automatización dentro de la compañía.

***Palabras clave* — Pruebas de software , Calidad de software, Automatización de pruebas, Control de calidad**

## ABSTRACT

This document presents the work carried out during the academic practices in which it was proposed to validate the quality of a CRM module using test automation tools and integrating these with the software development life cycle, seeking to improve the efficiency of testing processes also known as certification. The concepts related to this topic are explained, in which, among others, are DevOps, pipelines, CI/CD, Gherkin and in general how the tests were implemented working in an agile methodology similar to scrum, with the Webdriver IO framework, under the pattern POM in TypeScript, using a repository for version control, in addition to other knowledge acquired as a result of work such as automation to REST services. Finally, it is highlighted that in addition to meeting the objectives set at the beginning of the practices and the knowledge acquired and put into practice in this work, carrying it out has allowed supporting automation adoption processes within the company.

***Keywords* — Software tests, Software quality, Automation testing, Quality control**

---

## I. INTRODUCCIÓN

Durante el ciclo de vida del desarrollo de software, se encuentra la etapa de pruebas que es el proceso de evaluación y verificación de un producto o aplicación para saber si hace lo que se supone que debe hacer. Los beneficios de las pruebas incluyen la prevención de errores, la reducción de los costos de desarrollo y la mejora del rendimiento [1]. Muchas compañías de software tienen procesos de pruebas que se hacen de manera manual, se realizan en persona, haciendo clic a través de la aplicación o interactuando con el software y las API con las herramientas adecuadas. Por otro lado, se encuentran las pruebas automatizadas, que se realizan a través de una máquina que ejecuta un script de la prueba escrito con antelación [2] tanto en estas como en las manuales se ingresa, generalmente a un ambiente de pruebas o de calidad en el que se despliega el software desarrollado y se hacen las validaciones, luego de previamente haber realizado, entre otras actividades, un análisis y diseño de casos de pruebas, estos se desarrollan para definir las cosas que es necesario validar a fin de asegurar que el sistema funciona correctamente y está construido con un alto nivel de calidad [3]. Por lo general en la industria, se realizan ambos tipos de pruebas y todo el proceso es conocido como certificación.

En la célula de trabajo en la que estuvo el estudiante, se han realizado pruebas funcionales en un aplicativo CRM cuyo objetivo principal es mejorar la comunicación con los clientes. Los procesos de certificación para la salida a producción de los diferentes módulos del CRM habían sido realizados haciendo uso de pruebas manuales y si bien estas han sido útiles y efectivas para realizar dichos procesos, en muchas ocasiones pueden ser poco eficientes, por ejemplo, cuando se requieren realizar pruebas de regresión, para validar que nuevos cambios sobre el aplicativo no impacten las funcionalidades ya implementadas, se vuelven a ejecutar de forma manual una gran cantidad de las pruebas realizadas. Por otro lado, ocurre que muchos de los casos de prueba tienen pasos en común, es decir, en estos se requieren hacer acciones similares sobre el aplicativo lo que ocasiona que se realicen tareas repetitivas, esto sucede cuando se prueban módulos del CRM de forma individual o a nivel general entre los diferentes módulos, ya que a pesar de que son distintos a nivel funcional, son parecidos en cuanto a campos y elementos al utilizar la misma interfaz gráfica. Todo lo anterior afecta el time to market de las diferentes funcionalidades desarrolladas, lo que a su vez impacta en los costos de los proyectos.



En este informe, se presenta el trabajo realizado durante la práctica académica en la que se propuso la implementación de pruebas automatizadas al módulo de CRM con el objetivo de validar que cumpla con los requerimientos del usuario a la vez que se mejora la eficiencia en los procesos de certificación. Se presentan los términos relacionados con la automatización de pruebas, el enfoque que se siguió para cumplir con los objetivos que se hizo a través de la metodología Choucair, los resultados entre los que se encuentra el lograr por primera vez implementar automatización en las pruebas del CRM y las conclusiones a las que se llegaron. Es importante mencionar que debido a contratos de confidencialidad de la información y datos del cliente para el que se ha trabajado durante las prácticas académicas, no se puede compartir información con un alto nivel de detalle, por lo cual en algunos temas se habla de una forma más general y en otros se mostrarán ejemplos que permitan ilustrar algunos de los resultados obtenidos.

## II. OBJETIVOS

### *A. Objetivo general*

Validar la calidad del producto de software CRM, haciendo uso de herramientas de automatización de pruebas.

### *B. Objetivos específicos*

- Diseñar pruebas que evalúen de forma adecuada el estado del desarrollo respecto a la calidad del producto.
- Implementar los casos de prueba haciendo uso de herramientas de automatización.
- Integrar la automatización de pruebas en los procesos de CI/CD del desarrollo de software.

### III. MARCO TEÓRICO

CRM se refiere al conjunto de prácticas, estrategias comerciales y tecnologías enfocadas en la relación con el cliente. Contando con un sistema de CRM, las empresas de todos los tamaños pueden mantenerse conectadas con los clientes, optimizar los procesos, mejorar la rentabilidad e impulsar el crecimiento del negocio [4]. Las pruebas automatizadas propuestas al módulo del CRM consisten en la aplicación de herramientas de software para automatizar el proceso manual de revisión y validación de un producto de software que lleva a cabo una persona. Las pruebas automatizadas aportan enormes beneficios a la eficiencia del equipo y al ROI de los equipos de control de calidad [2], que influye a su vez en la reducción del time to market ya que es el tiempo que transcurre desde que proyectamos un producto y servicio y el tiempo que tarda en estar disponible para el consumidor final [5]. El trabajo realizado se hizo bajo una metodología similar a scrum que es un marco de gestión de proyectos de metodología ágil que ayuda a los equipos a estructurar y gestionar el trabajo mediante un conjunto de valores, principios y prácticas [6].

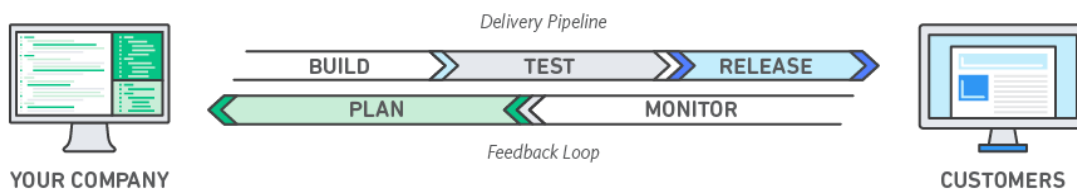


Fig. 1. Modelo de DevOps

Nota: fuente <https://aws.amazon.com/es/devops/what-is-devops/>

Las operaciones de desarrollo constituyen una combinación de filosofías culturales, prácticas y herramientas que incrementan la capacidad de una organización de proporcionar aplicaciones y servicios a gran velocidad: desarrollar y mejorar productos con mayor rapidez que las organizaciones que utilizan procesos tradicionales de desarrollo de software y administración de la infraestructura. Esta velocidad permite a las organizaciones servir mejor a sus clientes y competir de forma más eficaz en el mercado, en la figura 1, se puede ver a nivel general el modelo DevOps, en este los equipos de control de calidad y de seguridad también se integran

---

más con el desarrollo y las operaciones e intervienen durante todo el ciclo de vida de la aplicación [7]. Normalmente dentro de estos procesos se encuentra CI/CD que es un método para distribuir las aplicaciones a los clientes con frecuencia mediante el uso de la automatización en las etapas del desarrollo de aplicaciones. En concreto, el proceso de integración y distribución continuas incorpora la automatización y la supervisión permanentes en todo el ciclo de vida de las aplicaciones, desde las etapas de integración y prueba hasta las de distribución e implementación [8]. Un pipeline de CI/CD consiste en una serie de pasos que deben ejecutarse en orden para distribuir la versión nueva de un sistema de software [9]. Con la implementación de automatización de pruebas se busca precisamente lo anterior, que los procesos de pruebas se integren con el desarrollo.

El proyecto se implementó utilizando POM que es un patrón de diseño usado comúnmente para automatizar los casos de prueba. El objeto Page es una clase orientada a objetos que actúa como una interfaz para la página de la aplicación bajo prueba. La clase de página contiene la representación de elementos web y métodos para interactuar con elementos web. Mientras automatizamos los casos de prueba, creamos el objeto de estas clases de página e interactuamos con los elementos web llamando a los métodos de estas clases [10]. Los casos de pruebas se escribieron en Gherkin, este es un lenguaje de dominio específico legible por negocios. Es fácil de entender, ya que elimina los detalles de lógica y sintaxis de las pruebas de comportamiento. Utiliza un conjunto de palabras clave para que las pruebas sean más comprensibles. Las palabras clave más utilizadas son: Característica, Dado, Cuando, Entonces y muchas más [11]. El código de automatización, se gestionó con control de versiones también conocido como control de código fuente, que es la práctica de rastrear y gestionar los cambios en el código de software. Los sistemas de control de versiones son herramientas de software que ayudan a los equipos de software a gestionar los cambios en el código fuente a lo largo del tiempo [12] y el código se creó en un entorno de desarrollo integrado (IDE) que es una aplicación de software que ayuda a los programadores a desarrollar código de software de manera eficiente. Aumenta la productividad de los desarrolladores al combinar capacidades como editar, crear, probar y empaquetar software en una aplicación fácil de usar [13].

Debido a la complejidad del módulo de CRM también se realizaron pruebas a API que es un conjunto de definiciones y protocolos que se usa para diseñar e integrar el software de las aplicaciones [14], más concretamente a API REST. Cuando el cliente envía una solicitud a través de una API de REST, esta transfiere una representación del estado del recurso requerido a quien lo haya solicitado o al extremo. La información se entrega por medio de HTTP [15]

#### IV. METODOLOGÍA

Durante todo el proceso, se trabajó bajo una metodología ágil, en la cual se tenían sesiones diarias donde se informaba el avance del día anterior, lo que se iba a realizar y si se tenía algún impedimento. También se tenían reuniones semanales en las cuales se revisaban en muchas ocasiones dudas que se tuviesen de las funcionalidades a desarrollar y en un par de ocasiones se tuvieron retrospectivas en las cuales el equipo de trabajo comentó cómo se sentía y se hablaron de oportunidades de mejora.

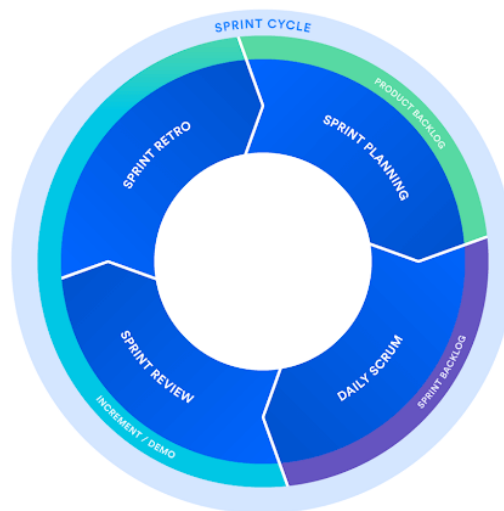


Fig. 2. Ciclo de un SPRINT en scrum

Nota: fuente <https://www.atlassian.com/es/agile/scrum>

Si bien no se trabajó bajo una metodología estrictamente scrum, se contaba con muchos de los elementos de dicha metodología, se podría decir que se adaptó al equipo de trabajo. En la

figura 2, se pueden ver algunos de dichos elementos de scrum, de estos como se comentó anteriormente, se contaba con un daily, retrospectivas, una planeación y dentro del equipo de trabajo estaba el scrum máster, PO y el equipo de desarrollo. Las reuniones semanales servían como reviews ya que en estas se compartían también los avances de la semana.

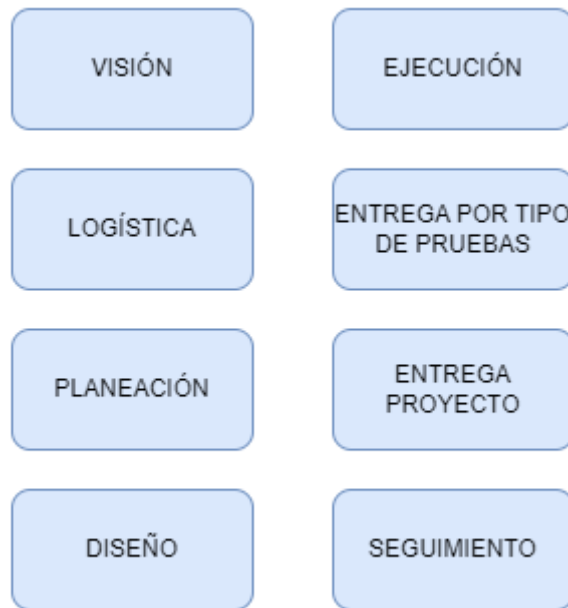


Fig. 3. Actividades realizadas en la práctica académica bajo metodología Choucair

Siguiendo la metodología de Choucair para pruebas de software, se realizaron las actividades que se muestran en la figura 3. En la visión se conoce al cliente, la necesidad que busca resolver con la solución a desarrollar, cómo se alinea con sus objetivos y cuál es la forma en la que se viene trabajando. En la logística se gestionan permisos, accesos, herramientas y documentación, esta etapa es transversal a todas las demás. En la planeación se crea el plan de pruebas que contiene el alcance, estrategia, los riesgos identificados en el proyecto, el cronograma y estimación del proyecto. Luego, tomando en cuenta la planeación, en el diseño se realiza el diseño de los casos de pruebas con el paso a paso a realizar, los resultados esperados y datos de pruebas y se analiza qué casos pueden ser automatizados. En la ejecución, se realiza la ejecución de los casos, la toma de evidencias y la gestión de bugs. En la entrega por tipo de prueba, se entrega el informe final de pruebas, lecciones aprendidas y conclusiones del proceso de ejecución de pruebas. En la entrega de proyecto, se realiza entrega del proyecto de todos los procesos de pruebas realizados, se actualiza base de conocimiento que se tenga y lecciones

---

aprendidas dentro del proyecto. Al final, en el seguimiento, se está pendiente de qué ocurre con el aplicativo o desarrollo puesto en producción. A continuación, se explica un poco más de lo realizado en cada una de las actividades de la metodología Choucair.

**Visión:** inicialmente en el proyecto, se obtuvo conocimiento respecto al cliente y la necesidad que se iba a resolver con la implementación del módulo del CRM y cómo esto está orientado con los objetivos estratégicos que tiene el cliente, se conoció cómo se venían trabajando en los diferentes procesos especialmente en la etapa de pruebas. Se tuvieron sesiones de explicación en las cuales se iba poco a poco comprendiendo mejor lo que se estaba construyendo. y como fruto de esto, se realizó la propuesta de prácticas en la cual se expuso al equipo de trabajo una demostración de automatización y se explicaron los beneficios que se tenía al implementar esta. Indagando más al respecto, con los diferentes compañeros de trabajo, se encontró que ya se había realizado automatización de pruebas en otros proyectos y se contaba con herramientas para poder implementarla. A raíz de lo anterior, se tuvieron unas sesiones de capacitación en estas tecnologías, lo positivo fue que como ya se tenía experiencia y conocimiento en automatización en Java y en desarrollo de software en el lenguaje de programación JavaScript, aprender el framework Webdriver IO en el lenguaje Typescript con el que se trabajó fue muy sencillo.

**Logística:** una de las ventajas que se tuvo para lograr los objetivos fue que se inició en una etapa muy temprana del desarrollo por lo cual se contaba con el suficiente tiempo para ir preparando los diferentes permisos y accesos a las herramientas y las configuraciones que suelen tener, además de que, a pesar de los beneficios que tiene la automatización, inicialmente el esfuerzo requerido es muy alto. Al inicio, se hizo solicitud de un repositorio para el versionamiento continuo del código en el cual se trabajó con git, se solicitó acceso a una herramienta para gestionar los datos de las pruebas y se configuró un proyecto en un IDE que serviría de partida para ir diseñando y creando las diferentes pruebas.

Una vez se contó con un ambiente de desarrollo, fueron gestionados los permisos para acceder allí e ir avanzando en la medida de lo posible en la implementación de las pruebas y también se realizaron pruebas exploratorias en las cuales se lograron encontrar muchos fallos que

fueron corregidos, lo cual fue muy positivo debido a que es mucho más barato hacer las correcciones en etapas tempranas del desarrollo [16]. Tal y como se había planteado el proyecto se tendría un MVP y hasta entonces no se pasaría a producción por completo el front, sin embargo, muchas partes del backend fueron desarrolladas y a estas también se les tuvo que realizar pruebas, lo cual también permitió conocer y aprender de otras herramientas para automatizar backend, principalmente a servicios REST.

Ya en la parte final de las prácticas, se gestionó la integración de las automatizaciones con el desarrollo de software, para ello se solicitaron permisos para tener pipelines, en los cuales la idea es en un futuro según sea el resultado de la ejecución automática de las pruebas, el proceso de despliegue continuo del código de desarrollo se ejecute o no.

**Planeación:** durante la planeación se delimitó el alcance de las pruebas, además de todo lo relacionado con el backend y front del nuevo módulo, se llegó a la conclusión de que se debían hacer pruebas de no afectación a los otros módulos del aplicativo, se generó la estrategia a seguir en la cual se planteó ir implementando los diferentes casos de prueba a medida de que se avanzaba en el diseño y comprensión, además se tuvo claridad respecto a que algunas partes del backend se irían enviando a certificación para su paso a producción antes de que el front por completo pasara por dichos procesos. Dentro de los riesgos identificados, se encontró la falta de claridad respecto a muchas de las funcionalidades a implementar, por lo cual se tuvieron sesiones de entendimiento, que facilitaron la comprensión del sistema. En esta etapa se creó el cronograma y la estimación de los diferentes tiempos para lograr cumplir con los objetivos planteados.

**Diseño, ejecución y entrega por tipo de prueba:** una vez se contaba con los conocimientos, permisos y accesos a las diferentes herramientas y el conocimiento de lo que se iba a desarrollar, se procedió con el diseño de los casos de prueba y los que eran factibles de automatizar, se fueron implementando. Cabe destacar, que dentro del proceso de implementación se realizaron ajustes al framework y patrón de diseño utilizado para automatización, que han sido propuestos como mejoras, principalmente con el fin de utilizar buenas prácticas de programación, como por ejemplo el utilizar los principios SOLID. También dentro del proceso han surgido nuevos ajustes en el alcance y nuevos desarrollos, por lo que se ha requerido ajustar la planeación

diseñar e implementar nuevos casos de prueba y por ello el proceso ha sido cíclico a pesar de no contar, como por ejemplo en scrum, con sprints. Es importante mencionar que aún se encuentra en desarrollo el aplicativo por lo cual se continúa el diseño, ejecución y entrega por tipo de prueba.

**Entrega de proyecto y seguimiento:** debido a lo explicado anteriormente, que el sistema continúa en desarrollo, estas actividades han sido realizadas de cara a los componentes del backend que han salido a producción, algunos de los cuales son utilizados por otros sistemas. En cuanto a estas etapas se han realizado propuestas de mejora debido a la implementación de automatización a servicios REST y dichos servicios han pasado exitosamente a producción y no han presentado inconvenientes. Se espera, una vez concluya el desarrollo del módulo del CRM y se cierre su certificación el poder profundizar más el trabajo en estas actividades.

## V. RESULTADOS

Antes de empezar a explicar los resultados, es importante mencionar que se lograron cumplir los objetivos planteados inicialmente en la propuesta de prácticas tal y como se explicará a continuación, es importante de nuevo reiterar, que debido a la confidencialidad de los datos e información del cliente, no se pueden presentar videos o capturas de pantalla de las automatizaciones implementadas al CRM, por lo cual se hablará de los diferentes logros obtenidos y se ilustrará con un ejemplo sencillo.

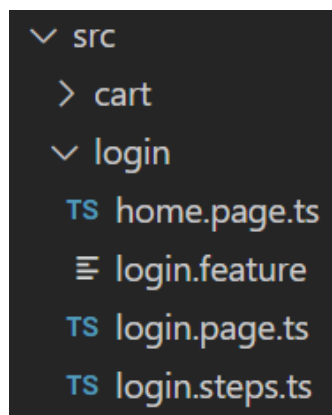


Fig. 4. Estructura de proyecto POM



En la figura 4, se puede observar la estructura de un proyecto POM similar al que se trabajó, en este, se encuentran carpetas que contienen las páginas (los archivos `.page.ts`), el feature (archivo `.feature`) y los steps (archivos `.steps`), como se puede ver, las extensiones de los steps y pages son `.ts` que es la extensión del lenguaje de programación TypeScript.

```
src > login > ≡ login.feature
1  @login
2  Feature: as an user I want to login
3
4    @successLogin
5    Scenario: success login
6      Given user was on sauce demo page
7      When enters username and password
8      Then should see home page title
```

Fig. 5. Ejemplo feature

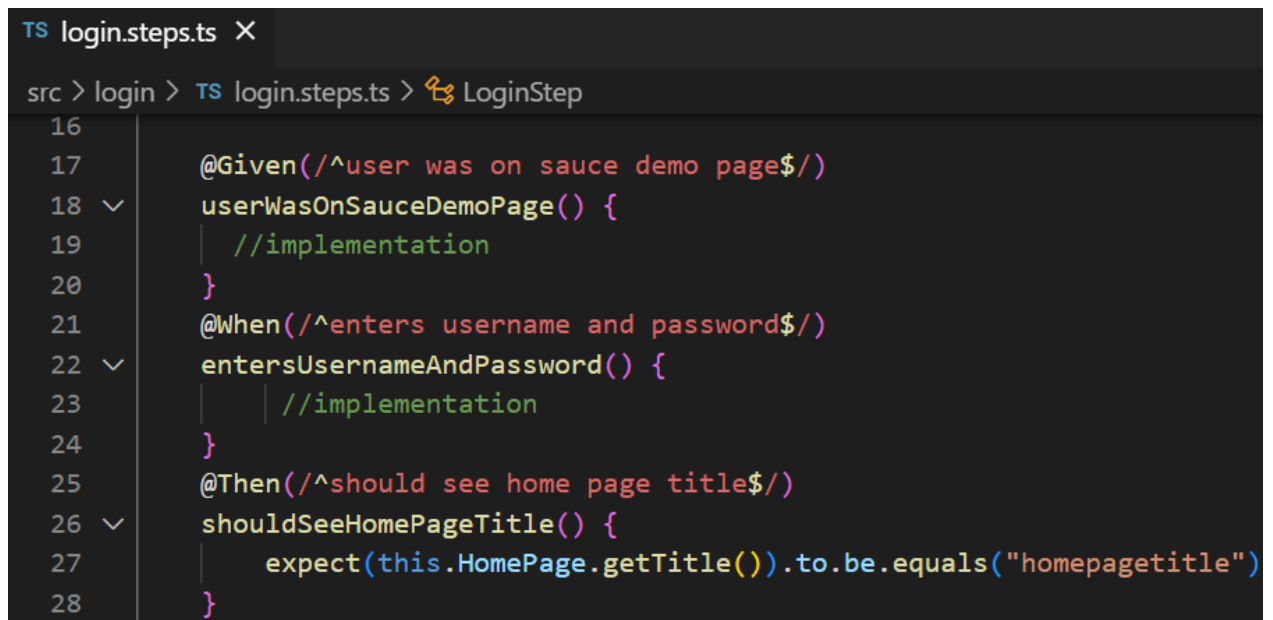
Una feature representa una historia de usuario, en la figura 5, se ilustra un ejemplo de una historia de usuario que podría ser un inicio de sesión en lenguaje Gherkin. Los `@tags` permiten facilitar la ejecución de las pruebas. Los escenarios representan los casos de prueba, para el ejemplo, un inicio de sesión exitoso, estos cuentan con las palabras Dado, Cuando, Entonces que describen el caso, el Dado habla de precondiciones, el Cuando de acciones que realiza el usuario y el Entonces es la validación o lo que se espera que ocurra al realizar las acciones. Un `.feature` muy parecido fue utilizado para automatizar el inicio de sesión en el CRM.

```
export class HomePage extends Page {
    private TITLE = $(".title")

    public getTitle(): string {
        return this.TITLE.getText()
    }
}
```

Fig. 6. Ejemplo clase Page

En la figura 6, se ilustra un ejemplo de una clase Page, que representa una página del aplicativo, en ellas se encuentran los elementos de la página, para el ejemplo el título y se acceden o realizan acciones sobre los elementos a través de los métodos de la página, que en este caso es obtener el texto del título.



```
TS login.steps.ts X
src > login > TS login.steps.ts > LoginStep
16
17 @Given(/^user was on sauce demo page$/)
18 userWasOnSauceDemoPage() {
19   //implementation
20 }
21 @When(/^enters username and password$/)
22 entersUsernameAndPassword() {
23   //implementation
24 }
25 @Then(/^should see home page title$/)
26 shouldSeeHomePageTitle() {
27   expect(this.HomePage.getTitle()).to.be.equals("homepagetitle")
28 }
```

Fig. 7. Ejemplo steps

En las steps se implementan los pasos de las feature, la idea es en estos llamar a los métodos de las páginas para interactuar con estas según los pasos. En la figura 7, se puede ver un ejemplo para los steps del inicio de sesión y en la línea 27 se muestra un ejemplo de la validación, en esta se espera que el título retornado por la page, sea igual a un valor determinado que en el ejemplo se quemó en el código, pero que para los casos del CRM se obtiene de un sistema a parte de gestión de datos de pruebas.

Con una estructura similar se creó el proyecto de automatización del módulo del CRM utilizando a su vez similares elementos, se logró diseñar las pruebas automatizadas utilizando Gherkin que ha permitido desde un lenguaje cercano a negocio, plantear las pruebas. Por primera vez en el cliente se lograron implementar pruebas automatizadas en el CRM que tal y como está compuesto, estas pueden servir de partida para que se implementen en otros módulos y debido a

que el aplicativo se encuentra en varias geografías, también podrían ser utilizadas no solo en Colombia. Se pudieron hacer procesos de certificación por primera vez en ambiente de calidad que incluyesen pruebas automatizadas a servicios REST, esto es muy importante de cara a que sirve de partida para que otros compañeros puedan también implementar dichas pruebas, que reducirían el tiempo de los procesos de certificación a mediano y largo plazo, ya que en muchos casos es requerido, cuando hay algún ajuste, el volver a ejecutar pruebas realizadas previamente, además que facilita la toma de evidencias al generarse de forma automática. Se logró avanzar en la integración de pruebas automatizadas con el ciclo de vida del desarrollo del software lo cual es muy positivo debido a que el cliente espera poder reducir los tiempos desde la ideación de un proyecto hasta su puesta en producción y la automatización juega un papel importante para poder lograr esto.

## VI. CONCLUSIONES

Se han logrado cumplir los objetivos planteados al inicio de la práctica académica, al implementar las pruebas automatizadas, integrarlas con los procesos de CI/CD de desarrollo, permitiendo validar la calidad del CRM. Todo este proceso ha permitido afianzar los conocimientos previos que se tenían de automatización y adquirir nuevos como por ejemplo el manejo del framework Webdriver IO bajo el patrón POM en TypeScript. Además de lo anterior, se aprendieron herramientas para automatización de pruebas a servicios REST que inicialmente no se habían considerado, pero que han permitido facilitar algunos de los procesos de pruebas como la toma de evidencias.

Ha sido crucial y esencial la formación recibida dentro de la universidad en temas tanto técnicos, por ejemplo el aprendizaje de las metodologías de trabajo ágil o pruebas de software, como de habilidades blandas, por ejemplo el trabajo en equipo o comunicación asertiva, para llevar a cabo de forma satisfactoria este proceso de prácticas. El enfrentarse a un proyecto de la industria es muy retador y lleva a tener que vivir experiencias, cambios y el tener que tomar decisiones que sin la formación recibida en la UdeA y el acompañamiento en este proceso, difícilmente hubiesen sido posibles.

Finalmente, gracias al trabajo realizado y lo aprendido en herramientas de automatización de pruebas, se han apoyado procesos de adopción y capacitación de estas dentro de los diferentes equipos de trabajo y de cara a los próximos meses, se espera continuar con dichos apoyos que también han permitido afianzar los conocimientos adquiridos y aportar desde otro lugar a la compañía.

## REFERENCIAS

- [1] IBM, "Prueba de software," [En línea]. Disponible en:  
<https://www.ibm.com/co-es/topics/software-testing#:~:text=La%20prueba%20de%20software%20es,y%20la%20mejora%20del%20rendimiento.>
- [2] Atlassian, "Prueba automatizada: La guía definitiva," [En línea]. Disponible en:  
[https://www.atlassian.com/es/continuous-delivery/software-testing/automated-testing.](https://www.atlassian.com/es/continuous-delivery/software-testing/automated-testing)
- [3] IBM, "Resumen de casos y suites de prueba," [En línea]. Disponible en:  
[https://www.ibm.com/docs/es/elm/6.0.3?topic=testing-test-case-test-suite-overview.](https://www.ibm.com/docs/es/elm/6.0.3?topic=testing-test-case-test-suite-overview)
- [4] Salesforce, "Estrategia de gestión de relaciones con los clientes (CRM)," [En línea]. Disponible en: <https://www.salesforce.com/mx/crm/>.
- [5] Connecting Visions Group, "Tiempo de comercialización: por qué importa," [En línea]. Disponible en: <https://connectingvisionsgroup.com/ideas/crecer-fidelizar/time-to-market/>.
- [6] Atlassian, "Scrum: Una guía práctica," [En línea]. Disponible en:  
<https://www.atlassian.com/es/agile/scrum#:~:text=%C2%BFQu%C3%A9%20es%20scrum%3F,de%20valores%2C%20principios%20y%20pr%C3%A1cticas.>
- [7] Amazon Web Services, "¿Qué es DevOps?," [En línea]. Disponible en:  
<https://aws.amazon.com/es/devops/what-is-devops/>.
- [8] Red Hat, "¿Qué es CI/CD?," [En línea]. Disponible en:  
[https://www.redhat.com/es/topics/devops/what-is-ci-cd.](https://www.redhat.com/es/topics/devops/what-is-ci-cd)
- [9] Red Hat, "¿Qué es un pipeline de CI/CD?," [En línea]. Disponible en:  
[https://www.redhat.com/en/topics/devops/what-cicd-pipeline.](https://www.redhat.com/en/topics/devops/what-cicd-pipeline)
- [10] GeeksforGeeks, "Modelo de objeto de página (POM)," [En línea]. Disponible en:  
<https://www.geeksforgeeks.org/page-object-model-pom/>.
- [11] GeeksforGeeks, "Métodos de prueba ágiles: Prueba impulsada por comportamiento (BDD)," [En línea]. Disponible en:  
[https://www.geeksforgeeks.org/agile-testing-methods-behavior-driven-testing.](https://www.geeksforgeeks.org/agile-testing-methods-behavior-driven-testing)
- [12] Atlassian, "¿Qué es el control de versiones?," [En línea]. Disponible en:  
[https://www.atlassian.com/es/git/tutorials/what-is-version-control.](https://www.atlassian.com/es/git/tutorials/what-is-version-control)

- [13] Amazon Web Services, "¿Qué es un IDE?," [En línea]. Disponible en:  
<https://aws.amazon.com/es/what-is/ide/>.
- [14] Red Hat, "¿Qué son las interfaces de programación de aplicaciones (API)?," [En línea].  
Disponible en:  
<https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>.
- [15] Red Hat, "¿Qué es una API REST?," [En línea]. Disponible en:  
<https://www.redhat.com/es/topics/api/what-is-a-rest-api>.
- [16] S. S. Miller, R. L. Cutler, y J. E. Dean, "Verification of Flight-Critical Systems: Formal Methods