Revista Ciencia & Tecnología

No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

Aplicación Web progresiva para la adopción de mascotas empleando la tecnología Angular-Ionic & CodeIgniter

Progressive Web application for the adoption of pets using Angular-Ionic & CodeIgniter technology

Danny Usca Farinango[1]
danny.usca@espoch.edu.ec
https://orcid.org/0000-0001-5332-0065
Diego Avila-Pesantez[2]
davila@espoch.edu.ec
https://orcid.org/0000-0001-8394-5621
George Figueras Benitez[3]
genriquef@usb.ve
https://orcid.org/0000-0001-8693-1217
Raúl Rosero Miranda[4]
raul.rosero@espoch.edu.ec
https://orcid.org/0000-0002-2315-9773

Recibido: 07/04/2021, Aceptado: 20/06/2021

Resumen

En la actualidad, una aplicación web progresiva permite la ejecución de su código en cualquier navegador compatible con los estándares de universalidad. En este sentido, este trabajo desarrolla una aplicación multiplataforma dedicada a la adopción y rescate de mascotas para la corporación "LADRA". En el desarrollo del aplicativo web y móvil se aplicó la metodología ágil SCRUM con ayuda de los frameworks Angular-Ionic y CodeIgniter basados en TypeScript y PHP respectivamente. Además, de un plan de pruebas definido en TestLink, que garantizó la creación de un software eficiente. Para evaluar la calidad del producto software se utilizó la métrica de Usabilidad, definida en la norma ISO/IEC 25010, aplicando una encuesta basada en el cuestionario USE de Arnold Lund; logrando medir las subcaracterísticas de inteligibilidad, operabilidad y estética. Al realizar un análisis descriptivo e inferencial se obtuvo valores positivos en la medición, dando como resultado que el aplicativo web tiene una interfaz agradable, útil, fácil de usar, satisface los requerimientos funcionales planteados y ahorra tiempo de proceso, siendo más productivos.

[1] Ingeniero de Sistemas Informáticos Escuela Superior Politécnica de Chimborazo. Facultad de Informática y Electrónica, Riobamba, Ecuador.

[2] Profesor Principal. Escuela Superior Politécnica de Chimborazo. Facultad de Informática y Electrónica, Riobamba, Ecuador.

[3] Profesor instructor. Universidad Simón Bolívar. Facultad de Electrónica. Caracas, Venezuela.

[4] Profesor Principal. Escuela Superior Politécnica de Chimborazo. Facultad de Informática y Electrónica, Riobamba, Ecuador.

Palabras clave: Ingeniería de Software, Aplicación web progresiva, LADRA, Usabilidad, TestLink.

Revista Ciencia & Tecnología
No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

Introducción

La forma exponencial como va creciendo la tecnología, hace que la sociedad comience a depender más de ella, y la acople en su diario vivir como una herramienta importante, para realizar sus diferentes actividades cotidianas (Rodríguez, 2017). Específicamente el uso de plataformas web y móviles son utilizadas para la mayoría de las actividades diarias, desde acceder a entretenimiento, estudio, noticias, libros, comercio electrónico hasta usarlas como herramientas indispensables para facilitar el trabajo (Jaimez-González, 2017). En la actualidad se puede observar varias plataformas web y móviles destinadas a la búsqueda y adopción de mascotas (ANIPAL, 2018; PetLife, 2018). Sin embargo, los aplicativos no están centralizados en brindar una solución integral utilizando herramientas mediante redes sociales.

La finalidad de la plataforma web y móvil denominada "LADRA" (Legión Activista por la Defensa y Respeto Animal) es compartir publicaciones de mascotas (perros y gatos) que se encuentren extraviadas, en adopciones o rescate en la ciudad de Riobamba, Ecuador. Para el desarrollo de dicho aplicativo se utilizó la metodología ágil SCRUM, que brinda un ambiente de buenas prácticas y garantiza mejores resultados. Conjuntamente, se utilizaron los frameworks Angular-Ionic, CodeIgniter, el gestor de base de datos MySQL y un plan de pruebas definido en TestLink. Finalmente, como parte del análisis de los resultados se utilizó el modelo de Calidad del Producto Software (ISO/IEC 25010) para evaluar la Usabilidad de la plataforma implementada.

A. Fundamentación

En las últimas décadas se han creado diferentes tipos de aplicaciones nativas, híbridas, web progresiva, y de página única, en beneficio del usuario, que se detallan en la Tabla 1. A continuación se describe las dos más importantes. *Aplicaciones Web Progresiva (PWA)* es similar a una aplicación web, que pueden ejecutarse en cualquier plataforma y dispositivos móviles (Tandel, Jamadar, & Technology, 2018). Además, una PWA tiene aspectos importantes que resaltan su uso, como el tamaño de instalación, tiempo de inicio y tiempo de renderizado de la barra de herramientas (Sheppard & Sheppard, 2017). Por otro lado, las *Aplicaciones de Página Única (SPA)* se establece dentro de las aplicaciones web, por lo que carga todos los recursos necesarios y no vuelve a actualizar la página mientras este en uso. El trabajo de todo este proceso se lo atribuye al navegador, por lo que utiliza el internet para su conexión y el servidor para almacenar información (Stefanović, 2017).

Para el óptimo desarrollo de este trabajo se ha seleccionado el uso de una aplicación web progresiva en la parte de desarrollo móvil, y una aplicación de página única para el aplicativo Web. En la tabla 1 se visualiza una comparativa de las principales aplicaciones.

Servicios Web RESTFUL - BackEnd

Contar con una correcta comunicación entre el FrontEnd y Backend es crucial al momento de desarrollar un aplicativo web o móvil, y es considerada como una pieza fundamental para seleccionar un framework.

No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

En este trabajo se escogió CodeIgniter, ya que contiene un núcleo potente y sencillo desarrollado para el lenguaje de programación PHP. Esto permite crear aplicaciones usando la arquitectura Modelo Vista Controlador (CodeIgniter, 2017). Una de sus características principales es su amplia gama de librerías, debido a ello lo hace ideal para configurar un servidor web RESTFUL en cuestión de minutos a diferencia de otros frameworks basados en PHP (Sidik, 2018). En comparación con otros frameworks para desarrollo de servicios web RESTFUL (ver Tabla 2), CodeIgniter cuenta con características positivas para la creación de un Backend potente.

Tabla 1. Características principales de las aplicaciones informáticas

Características	**Tipo de aplicación**				
	Nativa	**Híbrida**	**Web**	**PWA**	**SPA**
Presencia en tienda de aplicaciones	Si	Si	No	No	No
Velocidad	Muy rápida	Rápida	Rápida	Muy rápida	Muy rápida
Tamaño	Pesada	Pesada	Poco pesada	Ligera	Ligera
Disponible offline	Si	Si	No	Si	No
Interfaz de usuario	Buena	Buena	Regular	Buena	Buena
Multiplataforma	No	Si	Si	Si	Si
Posicionamiento SEO (Search Engine Optimization)	No	No	Si	Si	Si

Fuente: Jobe, 2013

Frameworks para FrontEnd

Angular es un framework de código abierto, que permite construir aplicaciones SPA mediante el modelo vista controlador (MVC) (Griffith, 2017). Además, Angular crea aplicaciones en el FrontEnd y brinda una experiencia de usuario excelente, utilizando bibliotecas basadas en JavaScript, y posee la división de código para efectuar una reutilización de objetos (Fain & Moiseev, 2017). Por otro lado, Ionic está basado en HTML5, CSS, JavaScript para crear aplicaciones multiplataformas, móviles, progresivas e hibridas, que brindan una experiencia similar a una aplicación móvil nativa. Además, proporciona el desarrollo de interfaces y aplicaciones complejas en un corto de tiempo reducido. En la tabla 3 se resalta sus características.

Revista Ciencia & Tecnología

No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

Tabla 2. Comparativa entre frameworks para desarrollo de servicios web RESTFUL

Características	Framework		
	Laravel	**Symfony**	**CodeIgniter**
Documentación	Amplia	Amplia	Amplia
Curva de aprendizaje	Alta	Media	Baja
Flexibilidad	No	Si	Si
Escalabilidad	Si	Si	Si
Seguridad Integrada	Si	No	Si
Arquitectura MVC	Si	Si	Si
Ideal para pequeños proyectos	No	No	Si

Fuente (Laaziri, Benmoussa, Khoulji, & Kerkeb, 2019)

Tabla 3. Comparativa entre frameworks para desarrollo

Característica	Framework			
	React Native	**Flutter**	**Xamarin**	**Angular-Ionic**
Curva de aprendizaje	Media	Alta	Alta	Media
Documentación	Amplia	Poca	Amplia	Muy amplia
Comunidad	Media	Baja	Media	Alta
Reusabilidad de código	Hasta un 90%	De 50% al 90%	Hasta un 96%	Hasta un 98%
Rendimiento móvil	Bueno	Bueno	Bueno	Muy bueno
Rendimiento web	Muy bueno	Pobre	Bueno	Excelente
Precio	Código abierto	Código abierto	Código abierto	Código abierto

Fuente: Isitan Koklu, 2017

ISO/IEC 25010

La Norma ISO/IEC 25010 (SQuaRE - Requisitos y evaluación de la calidad del software y del sistema) es un modelo para la evaluación de la calidad del producto software. Se maneja a través de características para evaluar las propiedades y los requisitos de los usuarios (Calabrese et al., 2017). La Usabilidad es la medida en

No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

que un producto pude ser utilizado por usuarios específicos para lograr objetivos con eficacia, eficiencia y satisfacción (Speicher, 2015).

TestLink

TestLink es una herramienta que permite crear y gestionar casos de pruebas basado en la web y posee una licencia Open Source. Por medio de los planes de prueba permite al grupo de trabajo ejecutar casos de test y registrarlos dinámicamente, de la misma manera genera informes y mantiene el objetivo de los requerimientos. Además, permite la gestión de actividades o proyectos de testing con la integración de herramientas como: bugtrakers, JIRA, entre otras (Cortés Pabón, 2020).

Materiales y métodos

Para el desarrollo del proyecto se utilizó la metodología ágil SCRUM, el cual tiene como base la creación de ciclos breves para el desarrollo, que comúnmente se llaman iteraciones conocidos como Sprints. De esta manera se trabajó directamente con el cliente y los entregables pasaron por un plan de pruebas mediante el uso del software TestLink. Dentro de la metodología ágil se ha definido las siguientes fases.

Product Backlog*:* Mediante reuniones mantenidas con personal que labora en la Corporación LADRA, se logró recabar información, que fue la recolección de los requerimientos, que fueron definidas en las historias de usuario (HU). Para dar valor de prioridad a las HU se asignan puntos de estimación, utilizando la técnica T-Shirt (ver Tabla 4) para la estimación de tiempos en el desarrollo de cada HU.

Tabla 4. Puntos estimados T-Shirt

Talla	Puntos Estimados	Horas-Trabajo
XS	5	5
S	10	10
M	12	12
L	20	20
XL	40	40

Fuente: elaboración propia

Sprint Backlog: En el proyecto se obtuvo un total de 37 HU y 8 historias técnicas divididas en los diferentes sprints, contabilizando un total de 15 sprints y cada uno tuvo un plazo de dos semanas para su entrega, trabajando un total de 40 horas de trabajo.

Desarrollo: Básicamente en esta fase se codificó los requerimientos recolectados para el desarrollo de la multiplataforma LADRA, además se definió la arquitectura del sistema, estándar de codificación, el diseño de la base de datos e interfaces de usuario y el plan de pruebas.

Revista Ciencia & Tecnología
No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

La arquitectura del sistema propuesto está basada en la comunicación de los componentes (web y móvil), se usa el patrón MVC (ver Figura 1) permitiendo separar la interfaz de usuario, la lógica de negocio y el acceso a datos, para tener ventajas en su desarrollo y estructura.

Sprint Backlog: En el proyecto se obtuvo un total de 37 HU y 8 historias técnicas divididas en los diferentes sprints, contabilizando un total de 15 sprints y cada uno tuvo un plazo de dos semanas para su entrega, trabajando un total de 40 horas de trabajo.

Desarrollo: Básicamente en esta fase se codificó los requerimientos recolectados para el desarrollo de la multiplataforma LADRA, además se definió la arquitectura del sistema, estándar de codificación, el diseño de la base de datos e interfaces de usuario y el plan de pruebas. La arquitectura del sistema propuesto está basada en la comunicación de los componentes (web y móvil), se usa el patrón MVC (ver Figura 1) permitiendo separar la interfaz de usuario, la lógica de negocio y el acceso a datos, para tener ventajas en su desarrollo y estructura.

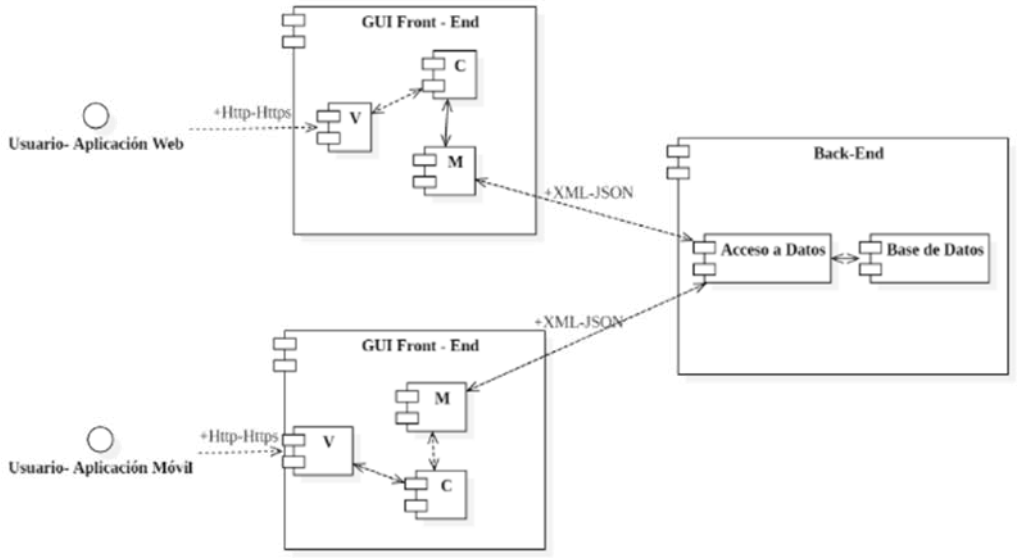

Figura 1. Arquitectura del sistema - Patrón MVC

El estándar de codificación para el desarrollo de los aplicativos fue LowerCamelCase, obteniendo una consistencia en la definición de métodos, atributos, variables, clases, entre otros. Esto facilitó la lectura de código y su mantenimiento, bajo buenas prácticas de programación. Complementando esta tarea, se diseñó de la base de datos (ver figura 2) enfocada en la persistencia de los datos e información de los aplicativos web y móvil de la Corporación "LADRA". En este proyecto se utilizó MySQL, normalizada hasta la tercera forma normal (3FN).

Revista Ciencia & Tecnología
No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

El diseño de las interfaces se realizó juntamente con el cliente, para llegar a un producto final. En base al logotipo se obtuvo un diseño minimalista para las diferentes pantallas. La maquetación se lo realizó con HTML basado en el sistema de mallado (Grid) y SASS (Syntactically Awesome StyleSheets) para los estilos, como se visualiza en la Figura 3 y Figura 4.

Mediante el uso de la herramienta TestLink se ejecutó un plan de pruebas y se creó diferentes suites, con casos de pruebas para la codificación de un software de calidad. De esta manera, se buscó cumplir el desarrollo de los requerimientos al tener un valor positivo frente a la Norma IEC/ISO 25010 - Usabilidad. El éxito o fracaso en la ejecución de un caso de prueba dependió de las precondiciones y pasos asignados. Una vez ejecutado el caso de prueba se guarda la información ingresada que generó un reporte (ver Figura 5), de esta manera se asegura dar una solución rápida en caso de ser necesario para obtener un software eficiente.

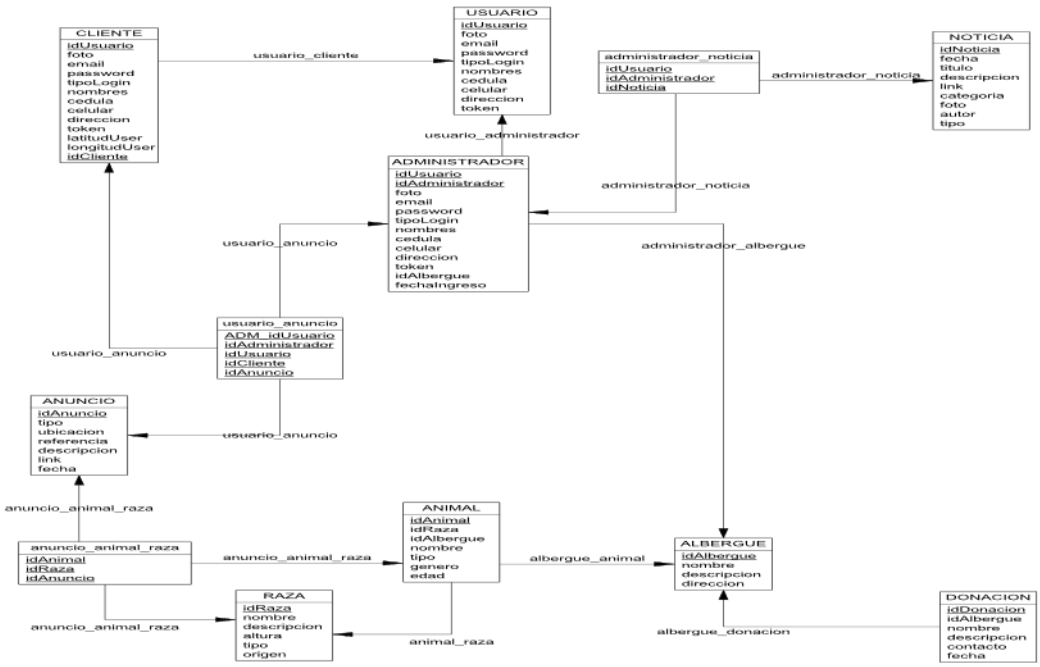

Figura 2. Diseño físico de la base de datos
Fuente: elaboración propia

Revista Ciencia & Tecnología
No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

Figura 3. Pantalla principal del aplicativo móvil. El mapa incorporado está bajo la licencia Open Source llamado Mapbox, el cual brinda servicios similares al conocido Google Maps.

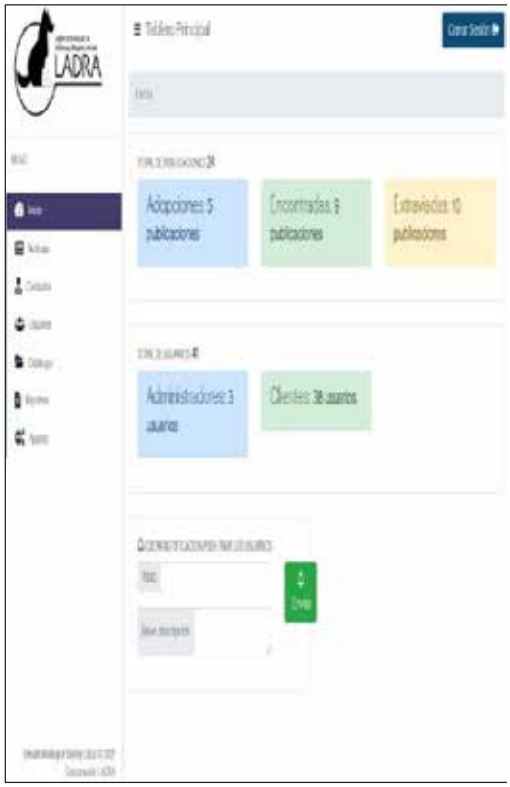

Figura 4. Pantalla principal de la aplicación web, predominando las pestañas: Panel de Control, Noticias, Usuarios, Reportes, Cuidados y Ajustes. Esto permite el acceso rápido hacia las funcionalidades del administrador.

El mapa incorporado de la figura 3 lo hace, como se aprecia, bajo la licencia Open Source (Mapbox), el que ofrece servicios semejantes al reconocido Google Maps. En cambio, en la figura 4 se ve que predominan las siguientes pestañas: Panel de Control, Noticias, Usuarios, Reportes, Cuidados y Ajustes. Esto permite el acceso rápido hacia las funcionalidades del administrador.

Finalización*:* La gestión del proyecto está reflejada en el Burn Down Chart (ver Figura 6), el cual demuestra un valor de total de 1200 puntos estimados divididos en un total de 15 sprints. Esta representación gráfica destaca el seguimiento del proyecto en realizar las funcionalidades a través del tiempo estimado para su desarrollo. Se puede visualizar el tiempo real (línea color rojo) y el tiempo estimado (línea color verde). El seguimiento se detalla a continuación.

El Sprint 3 requirió más tiempo debido a tener inconvenientes para subir imágenes al servidor, por lo que se creó un nuevo servicio web.

Revista Ciencia & Tecnología
No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

El Sprint 4 tuvo problemas con la versión del CLI de Ionic para cumplir con el requerimiento de recuperar credenciales, por el uso del correo electrónico. Los Sprints 6 y 8 presentaron desventajas en pruebas instalando la apk en un dispositivo físico. En el Sprint 15, los despliegues se vieron afectados por la versión del android-minSdkVersion (Antes nombrado: minSdkVersion) por actualizaciones de componentes de Ionic, se debe trabajar con un API superior a 28.

Caso de Prueba CL-1: Verificar que el usuario sea único [Versión : 1]

Autor: admin - 07/04/2021 09:06:52

Resumen:

Mediante el caso de prueba de verificar que el usuario sea único, se logrará controlar que no se guarde en la base de datos información repetida que pueda afectar al correcto funcionamiento de la plataforma

Precondiciones:

1. Tener descargado el aplicativo móvil en un smartphone
2. Tener conexión a internet
3. No estar registrado con el mismo correo
4. No estar registrado con el mismo número telefónico
5. No estar registrado con la misma cédula de identidad

Nº:	Pasos:	Resultados Esperados:	Notas de la ejecución:	Estado de la ejecución:
1	Tener descargado el aplicativo móvil en un smartphone	Visualizar la aplicación desde el dispositivo móvil	Todo correcto	Pasado
2	Ingresar a la aplicación móvil LADRA	Visualizar la interfaz principal de login	Todo correcto	Pasado
3	Dar clic en la opción de **Registrate aqui**	Visualizar la interfaz de Registro	Todo correcto	Pasado
4	Ingresar los nombres completos	Visualizar los nombres en la interfaz de registro	Todo correcto	Pasado
5	Ingresar la cédula	Visualizar la cédula en la interfaz de registro	Todo correcto	Pasado
6	Ingresar el número telefónico	Visualizar el numero telefónico en la interfaz de registro	Todo correcto	Pasado
7	Ingresar la dirección	Visualizar la dirección en la interfaz de registro	Todo correcto	Pasado
8	Ingresar un correo electrónico	Visualizar el correo electrónico en la interfaz de registro	Todo correcto	Pasado
9	Ingresar una contraseña	Visualizar la contraseña en la interfaz de registro	Todo correcto	Pasado
10	Dar clic en el botón de **Crear una Cuenta**	Visualizar un mensaje de aprobación en donde la cuenta ya este creada, y se redireccione a la pantalla de autentificación	Todo correcto	Pasado

Tipo de ejecución:	Manual
Duración estimada de la ejec. (min):	3.00
Importancia:	Alta
Requisitos	Ninguno
Keywords:	Ninguno
Detalles de la ejecución	
Tester	admin
Resultado de la Ejecución:	**Pasado**
Modo de Ejecución:	**Manual**
Duración de le ejecución (min):	**3.00**

Figura 5. Reporte generado al ejecutar un caso de prueba

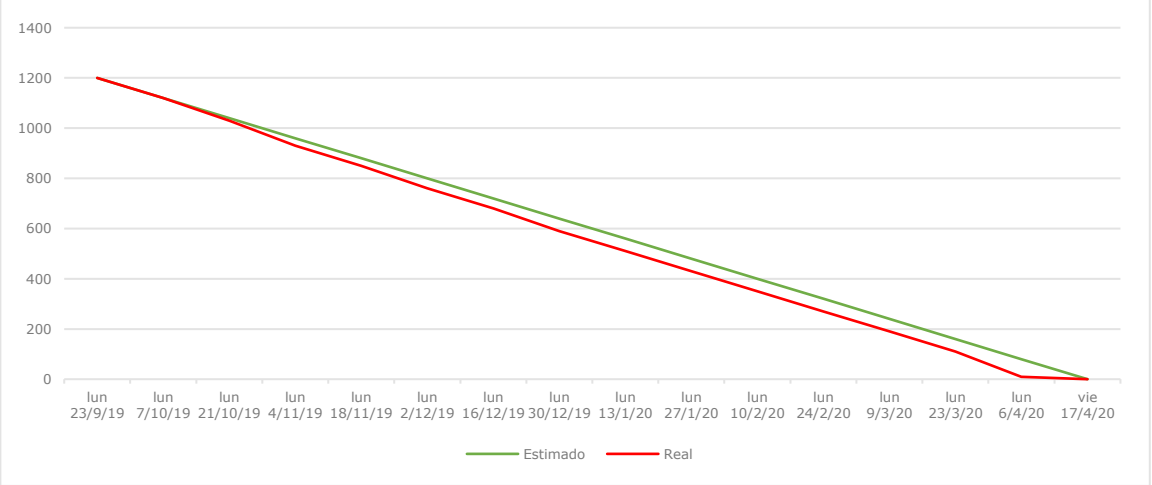

Figura 6. Burn Down Chart del Proyecto de desarrollo

Revista Ciencia & Tecnología

No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

Resultados y discusión

Para evaluar la plataforma se utilizó el estándar ISO/IEC 25010 y la Usabilidad como característica de calidad. Se adaptó el cuestionario USE (Lund, 2001) para aplicar a una población de 32 personas pertenecientes a la Corporación "LADRA". Para su valoración se utilizó una escala de Likert de 7 puntos, donde 1 es el valor más bajo y 7 el valor más alto, que evalúa tres subcaracterísticas de Usabilidad de la Norma ISO/IEC 25010: *inteligibilidad* (capacidad para reconocer su adecuación), *operabilidad* (capacidad para ser usado) y *estética* (de la interfaz de usuario). Al realizar el proceso de tabulación en cada una de las subcaracterísticas mencionadas se obtuvieron valores, que se muestra en la Tabla 5.

Tabla 5. Valores obtenidos al tabular las preguntas de cuestionario

	Inteligibilidad	**Operabilidad**	**Estética**
Muestra	32 personas		
Puntuación promedio estimada	16	20	12
Puntuación promedio obtenida	25,66	32,06	19,56
Desviación Estándar	1,52	1,54	0,98
Valor mínimo	21	29	16
Valor máximo	27	34	21
Valor ideal	28	35	21

Se procedió a realizar un análisis descriptivo e inferencial, por cada una de las subcaracterísticas de Usabilidad. La Inteligibilidad obtuvo una puntuación promedio de 25,66 superando por 9,66 a la puntuación promedio de 16. Para la operabilidad se obtuvo un valor de 32,06, superior a su puntuación estimada. Finalmente, la estética tuvo una puntuación promedio de 19,56 superando por 7,56 a la puntuación promedio de 12.

Para el análisis inferencial se aplicó la prueba de Shapiro-Wilk para verificar el tipo de distribución a la que pertenece (si es o no una distribución normal). Dando como resultado, que los datos en cada una de las subcaracterísticas no tienen una distribución normal, por lo tanto, se utilizó el test no paramétrico de Wilcoxon para contrastar sus medias. Para la Inteligibilidad se obtuvo un p-valor de 6,442e-07, siendo este resultado mucho menor a nivel de significancia de 0,05 (ver Figura 7). Por lo tanto, los usuarios establecen que la plataforma móvil es útil, satisface los requerimientos funcionales planteados y les ahorra tiempo de proceso, siendo más productivos.

Revista Ciencia & Tecnología
No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

```
> wilcox.test(data_inteligibilidad, mu = 16, exact = FALSE)

        Wilcoxon signed rank test with continuity correction

data:  data_inteligibilidad
V = 528, p-value = 6.442e-07
alternative hypothesis: true location is not equal to 16
```

Figura 7. Test de Wilcoxon para validar la Inteligibilidad

En la operabilidad, p-valor es igual a 7,14e-07, siendo este resultado mucho menor a nivel de significancia de 0,05 (ver figura 8). Por lo tanto, los usuarios establecen que la plataforma móvil es fácil de usar, lo pueden manipular sin necesidad de instrucciones, lo que conlleva a tener éxito cada vez que es usada y más importante realizan el trabajo con la menor cantidad de pasos.

```
> wilcox.test(data_operabilidad, mu = 20, exact = FALSE)

        Wilcoxon signed rank test with continuity correction

data:  data_operabilidad
V = 528, p-value = 7.145e-07
alternative hypothesis: true location is not equal to 20
```

Figura 8. Test de Wilcoxon para validar la operabilidad

En la subcategoría estética se calculó p-valor igual a 4,003e-07, siendo este resultado mucho menor a nivel de significancia de 0,05 (Ver Figura 9). Por lo tanto, los usuarios establecen que la plataforma móvil es agradable de usar, se encuentran satisfechos con sus características y se la recomendarían a sus amigos.

```
> wilcox.test(data_estetica, mu = 12, exact = FALSE)

        Wilcoxon signed rank test with continuity correction

data:  data_estetica
V = 528, p-value = 4.003e-07
alternative hypothesis: true location is not equal to 12
```

Figura 9. Test de Wilcoxon para validar la Operabilidad

Conclusiones

Para disminuir el tiempo de codificación en el sistema web progresivo se realizó un análisis de las diferentes herramientas disponible en el mercado. Como resultado, se seleccionaron dos frameworks Angular-Ionic y CodeIgniter, basados en los lenguajes de programación TypeScript y PHP respectivamente. Posterior al análisis de la tecnología, se utilizó la metodología ágil Scrum y un plan de pruebas en TestLink, que brindó resultados tempranos mediante las entregas continuas de funcionalidades y se logró una mayor productividad al controlar riesgos, y corregir errores en fases tempranas de desarrollo. Se obtuvo un total de 37 historias de usuario y 8 historias técnicas divididas en 15 sprints, construyendo un software acorde a los requerimientos funcionales establecidos con el usuario final.

No. 31, 31 de julio de 2021
ISSN impreso: 1390 - 6321
ISSN online: 2661 - 6734

Basado en los parámetros como la inteligibilidad, operabilidad y estética de la Norma ISO/IEC 25010 y por medio del análisis descriptivo e inferencial aplicando el USE Questionary, se obtuvo una puntuación promedio por cada ítem de usabilidad, para inteligibilidad un valor de 25,66 sobre 28, operabilidad un valor de 32,06 sobre 35 y estética un valor de 19,56 sobre 21. Los resultados sobrepasan la puntuación promedia establecida, concluyendo que la plataforma web y móvil tiene una interfaz agradable, útil, fácil de usar, satisface los requerimientos funcionales planteados y les ahorra tiempo de proceso, y mejora la productividad dentro de la corporación Ladra.

Referencias bibliográficas

ANIPAL. (2018). Retrieved from https://anipal.ec/

Calabrese, J., Muñoz, R., Pasini, A. C., Esponda, S., Boracchia, M., & Pesado, P. M. (2017). *Asistente para la evaluación de características de calidad de producto de software propuestas por ISO/IEC 25010 basado en métricas definidas usando el enfoque GQM.* Paper presented at the XXIII Congreso Argentino de Ciencias de la Computación (La Plata, 2017).

CodeIgniter. (2017). Codeigniter. com. In: Obtenido de https://www.codeigniter.com.

Cortés Pabón, Á. M. (2020). Automatización de pruebas de regresión para reducción de tiempo de entrega de nuevas versiones de software.

Fain, Y., & Moiseev, A. (2017). *Angular 2 Development with TypeScript*: Manning Publications Company.

Griffith, C. (2017). *Mobile App Development with Ionic, Revised Edition: Cross-Platform Apps with Ionic, Angular, and Cordova*: " O'Reilly Media, Inc.".

IŞITAN, M., & KOKLU, M. (2017). Comparison and Evaluation of Cross Platform Mobile Application Development Tools. *8*(4), 273-281.

Jaimez-González, C. R. J. R. R. I. p. l. I. y. e. D. E. (2017). Portal web con recursos didácticos digitales para el aprendizaje de HTML y CSS. *8*(15), 833-860.

Jobe, W. J. I. J. o. I. M. T. (2013). Native Apps vs. Mobile Web Apps. *7*(4).

Laaziri, M., Benmoussa, K., Khoulji, S., & Kerkeb, M. L. J. P. M. (2019). A Comparative study of PHP frameworks performance. *32*, 864-871.

PetLife. (2018). Retrieved from https://play.google.com/store/apps/details?id=com.comunadigital.petlife&hl=es_UY

Rodríguez, H. E. D. J. E. I. (2017). Tecnologías de la información y comunicación y crecimiento económico. *405*, 30-45.

Sheppard, D., & Sheppard, D. (2017). *Beginning progressive web app development*: Springer.

Sidik, B. (2018). FRAMEWORK CODE IGNITER 3.

Speicher, M. J. a. (2015). What is usability? a characterization based on ISO 9241-11 and ISO/IEC 25010.

Stefanović, M. (2017). DEVELOPMENT OF WEB BASED APPLICATION USING SPA ARCHITECTURE.

Tandel, S. S., Jamadar, A. J. I. J. o. I. R. i. S., Engineering, & Technology. (2018). Impact of progressive web apps on web app development. *7*(9), 9439-9444.