Ciencia Latina
Revista Multidisciplinar

DOI: https://doi.org/10.37811/cl_rcm.v6i4.2859

Python en desarrollo de aplicaciones de una sola página

José Boris Bellido Santa María
bellido.boris@usfx.bo
https://orcid.org/0000-0002-2380-7785
Universidad Mayor, Real y Pontificia de
San Francisco Xavier de Chuquisaca
Sucre – Bolivia

RESUMEN

Las aplicaciones de una sola página, en los últimos tiempos han permitido desarrollar aplicaciones que parezcan nativas, permitiendo de esta manera desarrollar y actualizar las aplicaciones de manera más eficiente.

El objetivo es contrastar el desarrollo de aplicaciones de una sola página con Python y con los marcos de trabajo más populares.

Se han aplicado diferentes métodos: Método del Análisis documental, para construir el sustento teórico. Método de la Medición, para definir las variables a estudiar. Método Experimental, se han desarrollado cuatro aplicaciones con la misma especificación de requerimientos usando Angular, React, Vue y PyScript.

Sobre el entorno de desarrollo, Angular, React y Vue fueron instalados con npm, dependen de cientos de paquetes para poder comenzar con el desarrollo de un proyecto, mientras que PyScript se puede usar por CDN y solo hace falta los archivos del proyecto. En cuanto al entorno de ejecución, Angular, React y Vue generan aplicaciones con tiempo de descarga muy parecidos, por otro lado PyScript tarda casi siete veces.

PyScript es una herramienta muy buena para personas que no conocen JavaScript, pero aún no está lista para ser usada en entornos de producción serios, en vista que el tiempo de descarga son demasiado alto.

Palabras clave: *aplicación una sola página; python; pyscript.*

Correspondencia: bellido.boris@usfx.bo
Artículo recibido: 05 agosto 2022. Aceptado para publicación: 15 agosto 2022.
Conflictos de Interés: Ninguna que declarar

Como citar: Bellido Santa María, J. B. (2022) Python en desarrollo de aplicaciones de una sola página. Ciencia Latina Revista Científica Multidisciplinar, 6(4) 3531-3552. DOI: https://doi.org/10.37811/cl_rcm.v6i4.2859

Python in single page app development

ABSTRACT

Single page applications, in recent times, have made it possible to develop applications that appear native, thus allowing applications to be developed and updated more efficiently.

The objective is to contrast the development of single page applications with Python and with the most popular frameworks.

Different methods have been applied: Documentary Analysis Method, to build the theoretical support. Measurement Method, to define the variables to study. Experimental method, four applications have been developed with the same requirements specification using Angular, React, Vue and PyScript.

Regarding the development environment, Angular, React and Vue were installed with npm, they depend on hundreds of packages to be able to start the development of a project, while PyScript can be used by CDN and only the project files are needed. Regarding the execution environment, Angular, React and Vue generate applications with very similar load times, on the other hand PyScript takes almost seven times.

PyScript is a very good tool for people who don't know JavaScript, but it's not quite ready to be used in serious production environments yet, as load times are too high.

Keywords: single-page-application; python; pyscript.

INTRODUCCIÓN

La web se ha convertido en los últimos tiempos en la cara de Internet, llegando al punto que muchas personas no saben cuál es la diferencia. Internet es la red de redes donde reside toda la información; la web es un subconjunto de Internet que contiene información a la que se puede acceder usando un navegador (Latorre, 2018). Servicios de Internet como ser el correo electrónico, transferencia de archivos, mensajería instantánea y otros han creado interfaces web para que el usuario pueda acceder de manera más sencilla. La web es un conjunto de documentos interconectados por enlaces de hipertexto, disponibles en Internet que se pueden comunicar a través de la tecnología digital, un hipertexto es la mezcla de textos, gráficos y archivos de todo tipo, en un mismo documento (Latorre, 2018). El hipertexto ha hecho posible que cualquier archivo en Internet sea accesible, cuando no es posible su visualización en el navegador se descarga para que el usuario pueda acceder a su contenido usando otros programas disponibles en su computadora.

Desde los inicios de la web, las aplicaciones que se han desarrollado han permitido a los usuarios un acceso a diversos servicios de manera sencilla.

> Las aplicaciones Web son cada vez más populares y su uso ha acaparado los ámbitos científico, cultural, académico, empresarial entre otros, y esto es debido a las múltiples ventajas que el usuario tiene respecto a los programas de escritorio. Entre otras, las ventajas que podemos mencionar son: sistema operativo multiplataforma, ejecutadas por cualquier dispositivo informático que tenga conexión a internet, no requiere de la instalación de programas solo un navegador, las copias de seguridad son almacenadas en los servidores, la información que se genera puede ser compartida de forma simultánea por varias personas, el espacio ocupado por los datos está a cargo del servidor y es de fácil uso (Molina Ríos et al., 2018).

Al momento de desarrollar un aplicación web el entorno se abstrae en gran medida reduciéndose al navegador, no interesa el hardware, ni el sistema operativo, ni los otros programas instalados en la computadora. El navegador es lo más importante, un buen navegador será capaz de presentar adecuadamente la información para que el usuario

pueda beneficiarse de la misma; es necesario que los navegadores cumplan con los estándares para que sin importar el navegador la experiencia de uso sea similar.

> El Consorcio World Wide Web (W3C) es una comunidad internacional en la que las organizaciones miembro, el personal a tiempo completo y el público trabajan juntos para desarrollar estándares web. Dirigido por el inventor y director de la Web Tim Berners-Lee y el CEO Jeffrey Jaffe, la misión del W3C es llevar la Web a su máximo potencial (World Wide Web Consortium, 2021).

Tanto las empresas que desarrollan navegadores como las empresas y personas que desarrollan aplicaciones web siguen los estándares emanados del W3C, el uso de estos estándares mejora la experiencia del usuario al momento de usar una aplicación web.

> Los estándares W3C definen una plataforma web abierta para el desarrollo de aplicaciones que tiene un potencial sin precedentes para permitir a los desarrolladores crear experiencias interactivas ricas, impulsadas por grandes almacenes de datos, que están disponibles en cualquier dispositivo. Aunque los límites de la plataforma continúan evolucionando, los líderes de la industria hablan casi al unísono sobre cómo HTML5 será la piedra Angular de esta plataforma. Pero toda la fuerza de la plataforma se basa en muchas más tecnologías que W3C y sus socios están creando, incluidas CSS, SVG, WOFF, la pila de Web Semántica, XML y una variedad de API.
>
> El W3C desarrolla estas especificaciones y pautas técnicas a través de un proceso diseñado para maximizar el consenso sobre el contenido de un informe técnico, para garantizar una alta calidad técnica y editorial y para obtener el respaldo del W3C y de la comunidad en general (World Wide Web Consortium, 2021).

El W3C propicia el encuentro y acuerdo entre los actores relacionados a la web, es decir los usuarios finales, los desarrolladores de páginas y aplicaciones web y las organizaciones dedicadas al desarrollo de navegadores. Cada navegador quiere conseguir mayor cuota del mercado, una tarea que solo es posible si los usuarios están satisfechos con la experiencia que ofrece cada navegador; por otro lado los desarrolladores web deben estar seguros que sus productos se presentarán de manera adecuada a los usuarios sin

importar el navegador que estén usando. El W3C marca el rumbo de la web para que las empresas puedan desarrollar navegadores que cumplen con lo mínimo para que los desarrolladores web puedan generar productos que los usuarios puedan usarlos sin problemas.

Las primeras aplicaciones web eran muy sencillas recogían los datos del usuario en un formulario, enviaban la información al servidor para ser procesadas, finalmente la respuesta era presentada en otra página web; el usuario debía esperar la culminación del proceso en cada interacción.

Las aplicaciones web cada vez son más complejas y poderosas, esto se debe en gran parte al aumento de las capacidades del hardware y de la velocidad de acceso a Internet. Según Speedtest, (2022), en Bolivia la velocidad de Internet, en el último año, se ha mantenido en una media de 25 Mbps en conexiones fijas, como se puede ver en las figura 1; con estas velocidades es posible que los usuarios puedan navegar y tener acceso a diferentes recursos y aplicaciones web sin problemas.

Toda aplicación web cuentan mínimamente con tres componentes: HTML siglas en inglés de HyperText Markup Language, en español Lenguaje de Marcado de Hipertexto; CSS siglas en inglés de Cascading Style Sheets, en español Hojas de Estilo en Cascada; y JavaScript.

HTML es el lenguaje para describir la estructura de las páginas web. HTML proporciona a los autores los medios para:

- Publica documentos en línea con encabezados, texto, tablas, listas, fotos, etc.
- Recupere información en línea a través de enlaces de hipertexto, con solo hacer clic en un botón.
- Diseñar formularios para la realización de transacciones con servicios remotos, para su uso en la búsqueda de información, realización de reservas, pedido de productos, etc.
- Incluya hojas de cálculo, videoclips, clips de sonido y otras aplicaciones directamente en sus documentos.

(World Wide Web Consortium, 2016).

HTML dispone de un conjunto de etiquetas que permiten definir tanto la estructura como el contenido. HTML permite definir la estructura y el contenido que se desea presentar al usuario; el contenido puede ser texto, imagen, sonido o vídeo que juntos ayudan a

comunicar de mejor manera el mensaje; además incluye hipervínculos o enlaces a otros documentos web y archivos que el usuario puede descargar. Por otro lado, HTML cuenta con la capacidad de recoger información del usuario a través de formularios para ser procesada en el servidor.

La estructura básica de un documento HTML está dada por las siguientes partes:

Encabezado: Normalmente formado por una gran franja que cruza la parte superior de la página con un gran título, un logotipo y quizás un lema. Esta parte suele permanecer invariable mientras navegas entre las páginas de un mismo sitio web.

Barra de navegación: Son los enlaces a las secciones principales del sitio web. Normalmente está formada por un menú con botones, enlaces o pestañas. Al igual que el encabezado, este contenido suele permanecer invariable en las diferentes páginas del sitio. Muchos diseñadores web consideran que el menú de navegación forma parte del encabezado y que no posee un componente individual, aunque no es necesario que sea así; de hecho, algunos argumentan que tener ambos componentes por separado es mejor por motivos de accesibilidad porque los lectores de pantalla pueden leer mejor ambos elementos si están separados.

Contenido principal: Es la gran parte central de la página y contiene la mayor parte del contenido particular de una página web concreta. ¡Esta es la parte que definitivamente debe variar mucho de una página a otra de tu sitio web!

Barra lateral: Suele incluir algún tipo de información adicional. Normalmente está relacionada con el contenido principal de la página, pero en otras ocasiones encontrarás elementos recurrentes como un menú de navegación secundario.

Pie de página: Es la parte inferior de la página, que generalmente contiene la letra pequeña, el copyright o la información de contacto. Es el sitio donde se coloca la información común, pero esta información no suele ser tan importante o es secundaria con respecto a la página en sí misma.

(Mozilla Corporation, 2021).

La adecuada definición de estas partes ayuda a que los usuarios puedan acceder de manera más fácil al contenido, en vista que el navegador podrá representar adecuadamente el contenido, sin importar si se trata de un navegador para personas con problemas visuales o entornos de consola.

> CSS es el lenguaje para describir la presentación de las páginas web, incluidos los colores, el diseño y las fuentes. Permite adaptar la presentación a diferentes tipos de dispositivos, como pantallas grandes, pantallas pequeñas o impresoras. La separación de HTML de CSS facilita el mantenimiento de sitios, el intercambio de hojas de estilo entre páginas y la adaptación de páginas a diferentes entornos. Esto se conoce como la separación de la estructura (o contenido) de la presentación (World Wide Web Consortium, 2016).

En segundo lugar, CSS se encarga de toda la parte visual de la web, a partir del contenido en el HTML, CSS da las instrucciones al navegador de cómo presentar cada uno de los elementos partiendo del tipo y formato de dispositivo que está consumiendo la aplicación web.

Con el uso intensivo de los teléfonos inteligentes y su capacidad de acceder a la web, es cada vez más frecuente que las personas usen las aplicaciones web desde el teléfono. Las computadoras, generalmente, tienen pantallas dispuestas horizontalmente, mientras que los teléfonos, en la mayoría de los casos, se usan de forma vertical; el mismo contenido se debe presentar de manera diferente según en tipo de dispositivo, el CSS tiene la responsabilidad de definir cómo presentar un contenido según el dispositivo que presenta la información.

Finalmente, el lenguaje de secuencias de comandos más común es ECMAscript (más conocido como JavaScript) es desarrollado por Ecma (World Wide Web Consortium, 2016); ECMAScript es un lenguaje de programación orientado a objetos para realizar cálculos y manipular objetos computacionales dentro de un entorno del cliente (Ecma International, 2021). JavaScript se ejecuta en el navegador, permite manipular tanto el HTML como el CSS para modificar el contenido y la apariencia del contenido presentado. Además permite que la aplicación web sea más interactiva, muchas tareas no necesitan procesar datos en el servidor. Evitar la consulta al servidor reduce los tiempos de espera del usuario, las acciones se ejecutan de manera rápida en el navegador.

> Un navegador web proporciona un entorno de cliente de ECMAScript para el cálculo del lado del cliente que incluye, por ejemplo, objetos que representan ventanas, menús, ventanas emergentes, cuadros de diálogo, áreas de texto, anclas, marcos, historial, cookies, entrada y salida. Además, el entorno cliente proporciona un medio para adjuntar código a eventos como cambio de enfoque, carga de página e imagen, descarga, error y cancelación, selección, envío de formulario y acciones del mouse. El código aparece dentro del HTML y la página que se muestra es una combinación de elementos de la interfaz de usuario y texto e imágenes fijos y calculados. El código es Reactivo a la interacción del usuario y no se necesita un programa principal (Ecma International, 2021).

La conjunción de HTML, CSS y JavaScript permiten presentar el contenido de manera adecuada al usuario y le dan la posibilidad de interactuar con el mismo. Conocer y entender las capacidades y limitaciones de los mismos son importantes para el desarrollo de aplicaciones web.

Para ayudar en el proceso de desarrollo existen en la industria muchos marcos de trabajo y bibliotecas que ayudan con las tareas relacionadas a la definición del CSS y a la programación con JavaScript. Si bien las empresas que desarrollan navegadores están comprometidas con implementar los estándares no siempre lo hacen oportunamente, por eso los marcos de trabajos generalmente añaden una capa que abstrae al navegador, permitiendo al desarrollador concentrarse en desarrollar las funcionalidades solicitadas y no a lidiar con las particularidades de cada navegador.

> Existen dos enfoques generales para crear aplicaciones web en la actualidad: las aplicaciones web tradicionales que realizan la mayor parte de la lógica de la aplicación en el servidor y las aplicaciones de una sola página (SPA) que realizan la mayor parte de la lógica de la aplicación en la interfaz de usuario, en un navegador web, comunicándose con el servidor utilizando principalmente API web. También es posible un enfoque híbrido, siendo más simple alojar una o más subaplicaciones tipo SPA dentro de una aplicación web tradicional más grande (Microsoft, 2022).

En las aplicaciones web tradicionales cada acción del usuario requiere que toda la página web sea generada nuevamente en el servidor y presentada al usuario en el navegador, la mayor parte del trabajo de programación se realiza en el lado; este enfoque permite trabajar con cualquier lenguaje de programación en vista que todos pueden generar HTML y entregarlo al usuario para que el navegador lo presente.

Por otro lado, las SPA siglas en inglés de Single-page application, en español Aplicación de una sola página, se genera una página al iniciar a la aplicación y no es necesario volver a cargarla. La estructura y el contenido de una SPA se actualiza de manera parcial en función de las interacciones que realiza el usuario. Una gran parte de la lógica de la aplicación se encuentra en el interfaz, se usa JavaScript para implementarla.

> Una SPA (aplicación de una sola página) es una implementación de aplicación web que carga solo un documento web y luego actualiza el contenido del cuerpo de ese documento único a través de las API de JavaScript, como XMLHttpRequest y Fetch, cuando se muestra contenido diferente.
>
> Por lo tanto, esto permite a los usuarios usar sitios web sin cargar páginas completamente nuevas desde el servidor, lo que puede generar mejoras en el rendimiento y una experiencia más dinámica, con algunas desventajas como el SEO, se requiere más esfuerzo para mantener el estado, implementar la navegación y lograr un rendimiento y monitoreo significativos (Mozilla Corporation, 2021).

Sin la necesidad de descargar más páginas la experiencia del usuario es más fluida y se asemeja, cada vez más, a una aplicación nativa del sistema operativo. Las capacidades de los navegadores modernos han permitido que no sea posible distinguir, a simple vista, si se trata de una aplicación web o una nativa.

Las SPA dependen de que los navegadores y de la versión de JavaScript que son capaces de ejecutar, en caso de que la SPA requiere una funcionalidad no disponible en el navegador es necesario usar o implementar una biblioteca que implemente las características faltantes. La verificación de funcionalidades, en los diferentes navegadores del mercado, puede ser un proceso muy costoso y distrae al equipo de su objetivo principal que es desarrollar funcionalidades que ayuden al usuario. Al igual que el trabajo con CSS, se han desarrollado muchos marcos de trabajo que permite usar

JavaScript sin preocuparse de la compatibilidad con los navegadores; estos marcos de trabajo además dan un conjunto de herramientas que permiten al equipo ser más productivo.

> Los marcos de trabajo para JavaScript son una parte esencial del desarrollo web front-end moderno, ya que brindan a los desarrolladores herramientas probadas para crear aplicaciones web escalables e interactivas. Muchas empresas modernas utilizan marcos como parte estándar de sus herramientas, por lo que muchos trabajos de desarrollo front-end ahora requieren experiencia en marcos (Mozilla Corporation, 2022).

La gran cantidad de herramientas que proporcionan los marcos de trabajo han hecho que los desarrolladores tengan que estudiar las mismas para obtener el mayor beneficio posible, si bien todos usan JavaScript como lenguaje de programación son tan diferentes que en algunos casos requieren mucho tiempo para dominarlos, por esto muchos desarrolladores se especializan y así ser más productivos.

Sobre los marcos de trabajo más relevantes, para Mozilla Corporation (2022) son muy populares y están bien establecidos: React, Ember, Vue y Angular, mientras que Svelte es un recién llegado comparativo que promete mucho y ha ganado mucha popularidad recientemente. Por otro lado, en las tendencias de Stack Overflow (2022), React, Angular y Vue ocupan las primeras posiciones, en ese orden, como se puede ver en la figura 2. El interés por estos marcos de trabajo muestra su relevancia en la industria del desarrollo de software ya sea para usarlos en nuevos proyectos o para dar mantenimiento a productos existentes. Para Potter (2022), la cantidad de descargas desde NPM muestra que React, Vue y Angular son los más populares, en ese orden, como se puede ver en la figura 3.

Wasm es un formato de instrucción binaria para una máquina virtual basada en pila; Wasm está diseñado como un objetivo de compilación portátil para lenguajes de programación, lo que permite la implementación en la web para aplicaciones de cliente y servidor (WebAssembly, 2020). Wasm permite romper la exclusividad de JavaScript a la hora de desarrollar funcionalidades que se ejecuten en el navegador, tener la capacidad de usar otros lenguajes de programación permite que otros profesionales pueden ayudar a desarrollar aplicaciones sin entrenamiento o capacitación; además se puede explotar

las capacidades de otros lenguajes al momento de plantear la solución a un problema dado.

WebAssembly (2020) se caracteriza por ser:

- Eficiente y rápido: La máquina de pila Wasm está diseñada para codificarse en un formato binario eficiente en tamaño y tiempo de carga. WebAssembly tiene como objetivo ejecutarse a velocidad nativa aprovechando las capacidades de hardware comunes disponibles en una amplia gama de plataformas.
- Abierto y depurable: WebAssembly está diseñado para imprimirse de forma bonita en un formato de texto para depurar, probar, experimentar, optimizar, aprender, enseñar y escribir programas a mano. El formato textual se utilizará cuando se visualice la fuente de los módulos Wasm en la web.
- Seguro: WebAssembly describe un entorno de ejecución en espacio aislado y seguro para la memoria que incluso puede implementarse dentro de las máquinas virtuales de JavaScript existentes. Cuando se incrusta en la web, WebAssembly aplicará las mismas políticas de seguridad de origen y permisos del navegador.
- Parte de la plataforma web abierta: WebAssembly está diseñado para mantener la naturaleza de la web sin versiones, con funciones probadas y compatible con versiones anteriores. Los módulos de WebAssembly podrán entrar y salir del contexto de JavaScript y acceder a la funcionalidad del navegador a través de las mismas API web accesibles desde JavaScript. WebAssembly también admite incrustaciones no web.

Para que Wasm sea una alternativa real a JavaScript debe ser igual de eficiente y rápido, en este sentido Wasm usa todos los recursos, de hardware, disponibles a los que puede acceder el navegador para dar al usuario la mejor experiencia posible, sin que perciba alguna diferencia en el rendimiento.

Wasm tiene mecanismos que permiten una depuración fácil del código, esta característica permite desarrollar las aplicaciones de mejor manera, pues es posible inspeccionar el código y encontrar los errores de forma más eficiente.

Respetar y usar las políticas y permisos del navegador genera un entorno seguro para la ejecución de las aplicaciones, donde los datos de los usuarios se encuentran a buen

recaudo; lo cual permite la implementación de funcionalidades donde la privacidad es un aspecto importante y crítico.

Tener acceso a las API disponibles actualmente por JavaScript permite usar los grandes avances de los últimos años en el desarrollo de aplicaciones web, no es necesario construir nuevas API para gozar de los beneficios con los que se cuenta en la actualidad.

Como se puede ver en la tabla 1, el apoyo por parte de la industria es importante, los principales navegadores, tanto para computadoras de escritorio como para dispositivos móviles están implementando y dando soporte a Wasm , incluyendo en sus ciclos de desarrollo características que les permitan cumplir con la especificación de Wasm.

Según LibHunt (2022), Pyodide es el mejor proyecto de código abierto para ejecutar Python sobre Wasm, con cerca de nueve mil estrellas en GitHub. Pyodide fue creado en 2018 por Michael Droettboom en Mozilla (Pyodide, 2021).

> Pyodide es una distribución de Python para el navegador y Node.js basada en WebAssembly/Emscripten.
>
> Pyodide permite instalar y ejecutar paquetes de Python en el navegador con micropip. Se admite cualquier paquete de Python puro disponible en PyPI. Muchos paquetes con extensiones C también han sido portados para su uso con Pyodide. Estos incluyen muchos paquetes de uso general, como regex, pyyaml, lxml y paquetes de Python científico, incluidos numpy, pandas, scipy, matplotlib y scikit-learn.
>
> Pyodide viene con una robusta interfaz de función externa JavaScript - Python para que se pueda mezclar libremente estos dos lenguajes en el código con una fricción mínima. Esto incluye soporte completo para el manejo de errores (lanzar un error en un lenguaje, detectarlo en el otro), async/await y mucho más.
>
> Cuando se usa dentro de un navegador, Python tiene acceso completo a las API web. (Pyodide, 2021).

La capacidad de usar módulos escritos con Python puro abre un universo de posibilidades, en vista de que Python es uno de los lenguajes más populares actualmente. Por otro lado, la capacidad de Wasm de compilar C/C++ permite tener acceso a extensiones desarrolladas con esta tecnología.

Pyodide, permite usar JavaScript y Python en las dos direcciones, es decir se puede ejecutar código escrito en Python desde JavaScript o viceversa, dando al desarrollador la posibilidad de usar lo mejor de los dos lenguajes.

Pyodide, incluye 102 paquetes (Pyodide, 2021), para diferentes propósitos; lo cual permite a los desarrolladores contar con un conjunto de poderosas herramientas para afrontar los proyectos.

PyScript fue propuesto por Anaconda Inc. en la PyCon US 2022 como una opción fácil de usar del lado del cliente, usando las capacidades de Wasm para ejecutar código escrito en Python en un navegador web.

> PyScript es un marco de trabajo que permite a los usuarios crear aplicaciones ricas de Python en el navegador utilizando la interfaz de HTML y el poder de Pyodide, WASM y las tecnologías web modernas. El marco PyScript brinda a los usuarios de todos los niveles de experiencia acceso a un lenguaje de programación expresivo y fácil de aprender con innumerables aplicaciones (PyScript, 2022).

PyScript es más sencillo de usar para los desarrolladores que tienen más experiencia con Python, Chiu, (2022) sostiene:

> ¡PyScript proporciona una API limpia y simple! Por ejemplo, puede usar <py-script src="/todo.py"></py-script> para cargar código escrito en Python. ¡Incluso puede importar otro script de Python, como lo hace normalmente! Mientras en Pyodide, se debe usar pyodide.runPython para ejecutar el código escrito en Python. También es muy conveniente si solo se quiere enviar el resultado a un elemento HTML mediante <py-script output="altair"></py-script>, donde altair es el id de un elemento HTML.

La necesidad de escribir código de JavaScript obligatorio se reduce a la mínima expresión, permitiendo desarrollar aplicaciones usando sólo Python. Sin embargo cuando es necesario es posible usar JavaScript de la misma manera que Pyodide.

Los principales componentes de PyScript son:

> Python en el navegador: Habilita el contenido directo, el alojamiento de archivos externos y el alojamiento de aplicaciones sin depender de la configuración del lado del servidor.

Ecosistema de Python: Ejecuta muchos paquetes populares de Python y la pila científica (como numpy, pandas, scikit-learn y más).

Python con JavaScript: Comunicación bidireccional entre objetos y espacios de nombres de Python y Javascript.

Gestión del entorno: Permite a los usuarios definir qué paquetes y archivos incluir para que se ejecute el código de la página.

Desarrollo de aplicaciones visuales: Usa componentes de interfaz de usuario seleccionados fácilmente disponibles, como botones, contenedores, cuadros de texto y más.

Marco de trabajo flexible: Un marco de trabajo flexible que se puede aprovechar para crear y compartir nuevos componentes conectables y extensibles directamente en Python.

(PyScript, 2022).

PyScript ha crecido rápidamente en poco tiempo llegando a tener más de trece mil estrellas en GitHub con una tasa de crecimiento del 13.1% (LibHunt, 2022).

El objetivo es contrastar el desarrollo de aplicaciones de una sola página con Python y con los marcos de trabajo más populares.

METODOLOGÍA

Método del Análisis documental: Se ha realizado la consulta bibliográfica a fuentes teóricas para caracterizar las aplicaciones de una sola página. Así mismo, a partir de la revisión documental se han identificado los marcos de trabajo más populares para el desarrollo de aplicaciones de una sola página.

Método de la Medición: Para poder comprar adecuadamente Python y los marcos de trabajo más populares se han definido dos grupos de variables:

- Entorno de desarrollo:
  - Cantidad de paquetes dependientes.
  - Tamaño de paquetes dependientes.
  - Cantidad de archivos del código fuente de la aplicación.
  - Tamaño de los archivos del código fuente de la aplicación.
  - Cantidad de archivos de todo el proyecto.
  - Tamaño de todo el proyecto.

- Entorno de ejecución:
  - Cantidad de archivos desplegables.
  - Tamaño de los archivos desplegables.
  - Tamaño de la descarga.
  - Tiempo de descarga.

Método Experimental: Se han desarrollado cuatro aplicaciones de una sola página usando los tres marcos de trabajo más populares actualmente, es decir Angular, React y Vue; y por otro lado Python con PyScript. Las aplicaciones desarrolladas resuelven el mismo problema, es decir obtener el clima de una determinada ciudad en grados Celsius o Fahrenheit. Las aplicaciones incorporan un formulario para obtener los datos por parte del usuario; la consulta a una API para conseguir los datos del clima; y la presentación de la información requerida. Las aplicaciones siguen la siguiente especificación de requerimientos:

Historias de usuario

Se utilizan las siguientes historias de usuario:

- Como usuario, quiero indicar una ciudad, para conocer el clima actual de dicha ciudad.

Criterios de aceptación:

  - Introducir por teclado.
  - No importa mayúsculas ni minúsculas.

- Como usuario, quiero definir una unidad de temperatura, para conocer el clima actual en dicha unidad.

Criterios de aceptación:

  - Elegir entre grados Celsius y Fahrenheit.
  - La opción por defecto debe ser Celsius.

- Como usuario, quiero datos del clima, para conocer el clima actual.

  Criterios de aceptación:

  - Los datos del clima se deben obtener de WeatherAPI.com.
  - Debe mostrar el nombre de la ciudad.
  - Debe mostrar el país al que pertenece la ciudad.
  - Debe mostrar la Temperatura actual.
  - Debe mostrar la Sensación térmica.

- Debe mostrar la Humedad.
- Debe mostrar la imagen del clima.
- Debe mostrar la descripción del clima.
- Debe mostrar la Hora local

Diseño de la Interfaz gráfica de usuario

El diseño de la interfaz gráfica de usuario usado se puede ver en la figura 4.

RESULTADOS Y DISCUSIÓN

Las cuatro aplicaciones desarrolladas son iguales para el usuario, se ven como en la figura 5, cumplen con todos los elementos de la especificación de requerimientos, la única diferencia se encuentra en las herramientas utilizadas para su desarrollo.

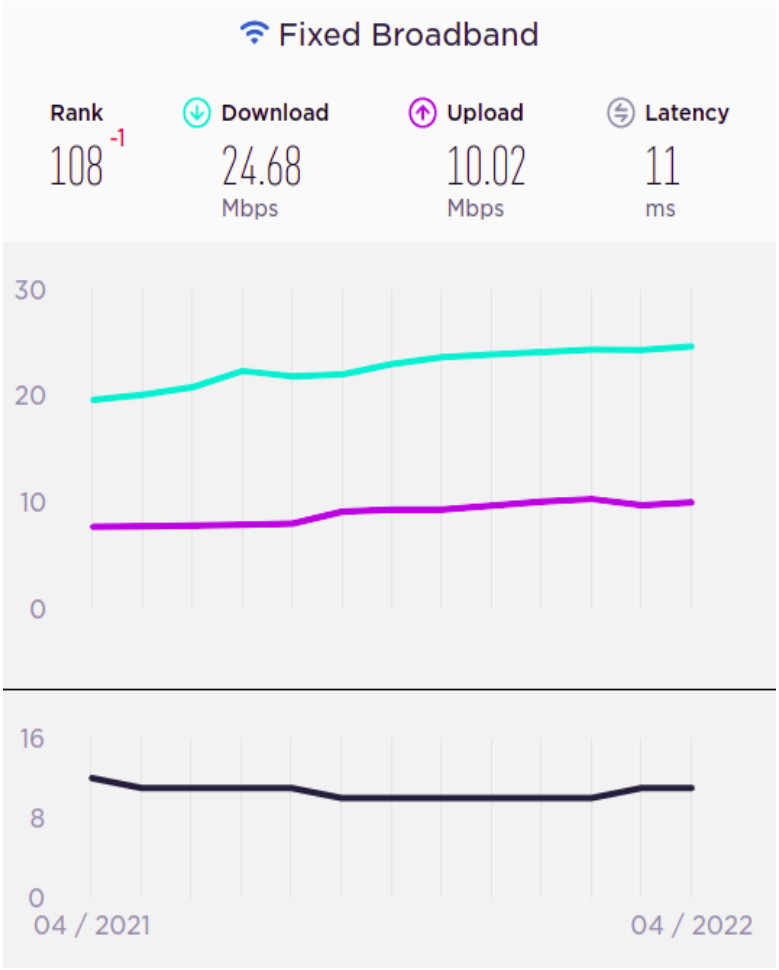
Angular, React y Vue se instalaron usando npm según las instrucciones y recomendaciones de cada herramienta, se utilizó el ambiente de desarrollo por defecto; en el caso de PyScript se utilizó la versión en línea disponible a través de CDN. En la tabla 2 se pueden ver las variables del entorno de desarrollo, donde se puede observar que los marcos de trabajo Angular, React y Vue necesitan de la instalación de cientos de paquetes para empezar a desarrollar de un proyecto; por otro lado PyScript al poder usarse a través de CDN no necesita la instalación de ninguna dependencia para tener el entorno de desarrollo listo. Cada herramienta requiere un conjunto de archivos para gestionar las diferentes partes del proyecto, Angular es la herramienta que más archivos necesita, usando 23 archivos; con respecto a PyScript se necesitaron cuatro archivos para la misma aplicación. Angular necesita más espacio en disco para el entorno de desarrollo, que incluyen las dependencias, el marco de trabajo y la aplicación; mientras PyScript requiere la menor cantidad de espacio gracias al uso por CDN.

Las aplicaciones desarrolladas con Angular, React y Vue para ser usadas por los usuarios finales requieren la construcción de un desplegable para su ejecución, este proceso permite convertir el código fuente de la aplicación en una versión que solo requiere de JavaScript para funcionar correctamente. En la tabla 3 se pueden ver las variables del entorno de ejecución, Angular genera 10 archivos para ser desplegados, mientras que Pyscript necesita cuatro, los mismo que corresponden al código fuente. React genera el proyecto desplegable que requiere más espacio en el servidor, en contraposición PyScript

requiere el menor espacio. Sin importar el número de archivos para el despliegue y el tamaño de los mismos, es muy importante saber cuántos kB necesita descargar el navegador del usuario para usar correctamente la aplicación, PyScript por lejos genera la aplicación que necesita más kB para renderizar correctamente la aplicación, por otra parte Angular es el más liviano. Finalmente, el tiempo de respuesta es sin duda un aspecto que mejora la experiencia del usuario, PyScript requiere demasiado tiempo, tarda casi siete veces más que las aplicaciones desarrolladas con Angular, React y Vue.

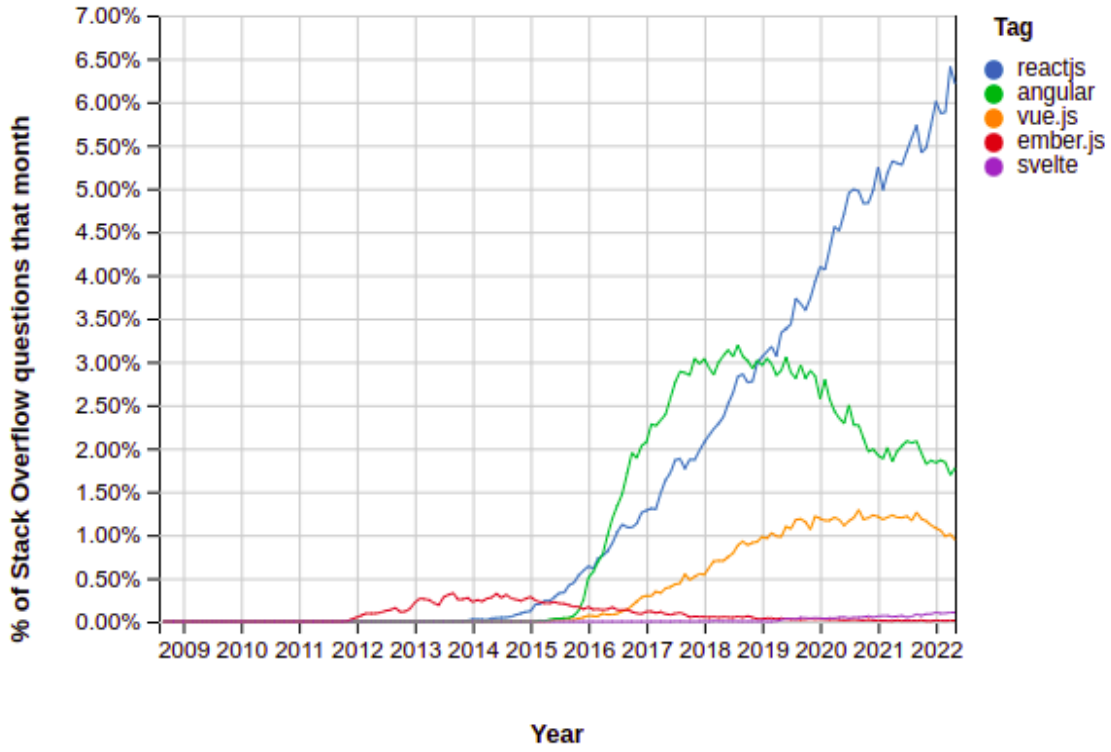
ILUSTRACIONES, TABLAS, FIGURAS

Figura 1. *Velocidad fija.*



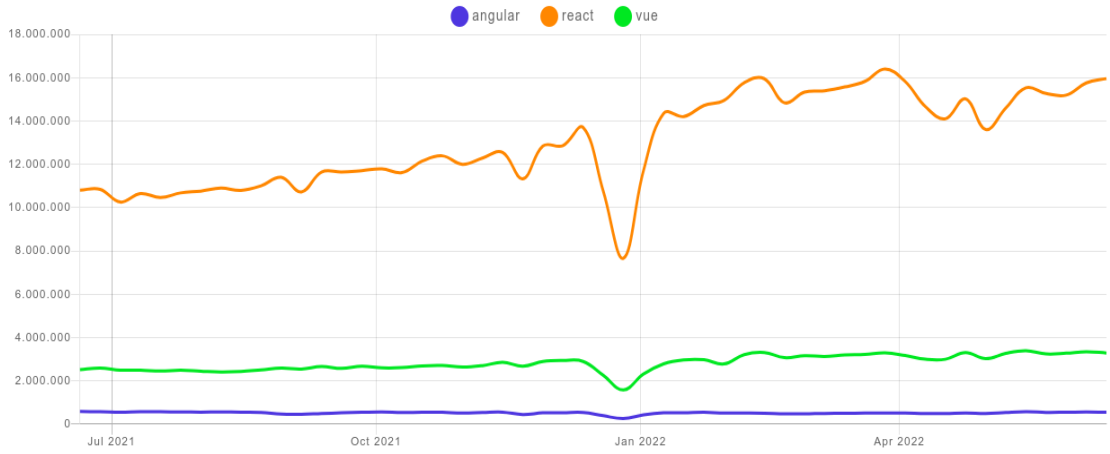
Fuente: Speedtest (2022).

Figura 2. *Tendencias de búsquedas.*



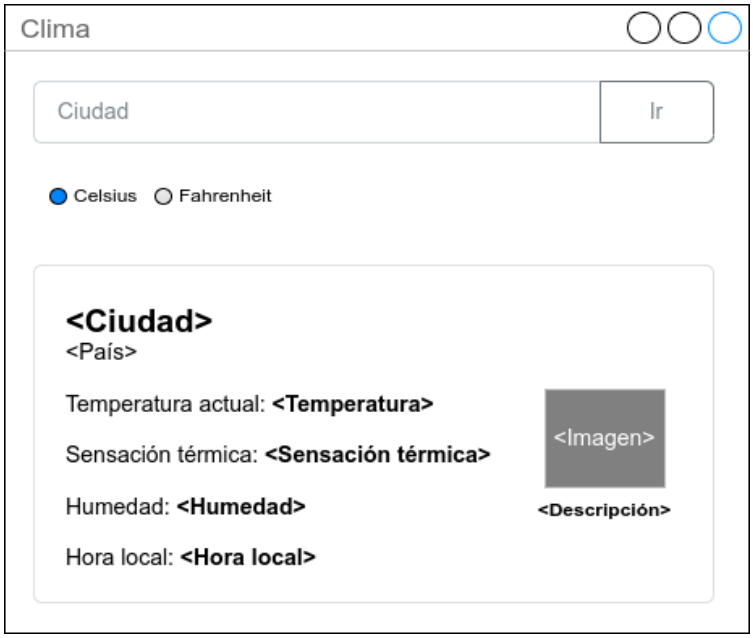
Fuente: Stack Overflow (2022).

Figura 3. *Descargas de npm.*



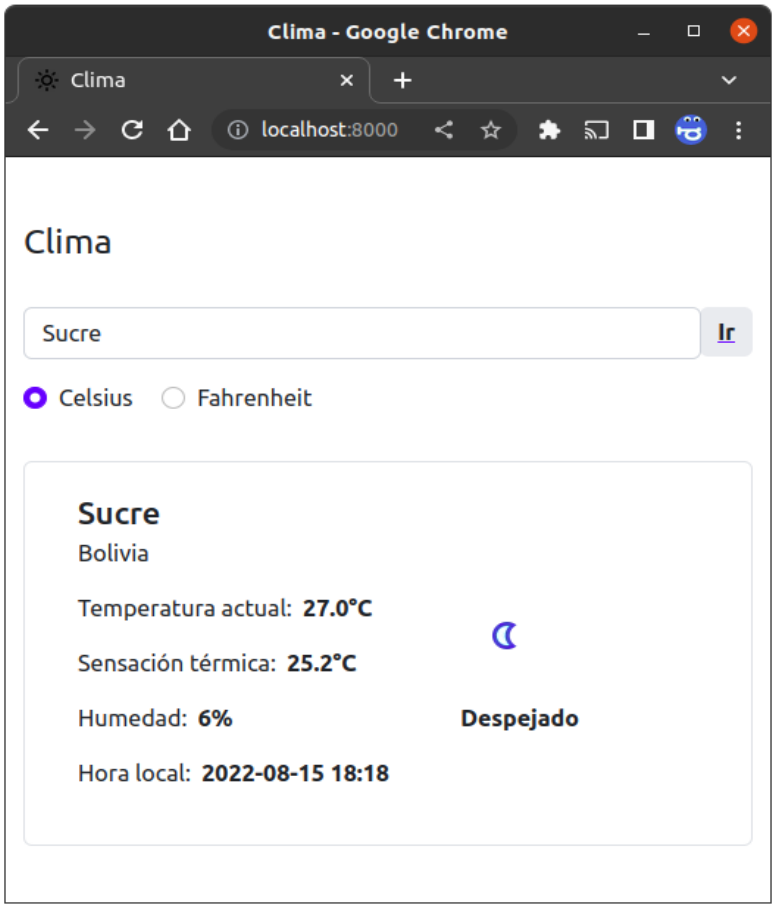
Fuente: Potter (2022).

Figura 4. *Diseño de la interfaz gráfica de usuario.*



Fuente: Elaboración propia.

Figura 5. *Aplicación resultante*.



Fuente: Elaboración propia.

Tabla 1. *Compatibilidad de navegadores con Wasm.*

Navegadores de escritorio	Navegadores móviles
▪ Chrome. ▪ Edge. ▪ Firefox. ▪ Opera. ▪ Safari.	▪ Chrome Android. ▪ Firefox for Android. ▪ Opera Android. ▪ Safari on iOS. ▪ Samsung Internet. ▪ WebView Android.

Fuente: Mozilla Corporation (2020).

Tabla 2. *Resultados del entorno de desarrollo*.

Variable	Angular	React	Vue	PyScript
Cantidad de paquetes dependientes.	588	791	654	0
Tamaño de paquetes dependientes.	276.2 MB	98.8 MB	33.8 MB	0
Cantidad de archivos del código fuente de la aplicación.	23	12	17	4
Tamaño de los archivos del código fuente de la aplicación.	31.2 kB	48.5 kB	34.3 kB	9.5 kB
Cantidad de archivos de todo el proyecto.	43165	17649	663	4
Tamaño de todo el proyecto.	277.3 MB	100.5 MB	42.1 MB	9.5 kB

Fuente: Elaboración propia.

Tabla 3. *Resultados del entorno de ejecución.*

Variable	Angular	React	Vue	PyScript
Cantidad de archivos desplegables.	10	7	4	4
Tamaño de los archivos desplegables.	206.8 kB	853.1 kB	67.6 kB	9.5 kB
Tamaño de la descarga.	32.49 KB	1.81 MB	315.17 KB	8.8 MB
Tiempo de descarga.	1.91 seg	1.95 seg	2.09 seg	13.15 seg

Fuente: Elaboración propia.

CONCLUSIONES

PyScript es una herramienta muy prometedora, pero aún no está lista para ser usada en entornos de producción serios, si bien su instalación y uso es sencillo, si se tiene experiencia con Python y desarrollo web; el tiempo de descarga y ejecución de la aplicación en el navegador es exageradamente alto, tanto en desarrollo como en producción. Si bien se tienen herramientas y mecanismos para una depuración efectiva, el tiempo de descarga de la aplicación reduce el ritmo del desarrollador.

LISTA DE REFERENCIAS

Chiu, P. (2022). *PyScript Vs. Pyodide: Which Should You Use?* https://python.plainenglish.io/pyscript-or-pyodide-f0f1dc10291f

Ecma International. (2021). *CMA-262*. https://262.ecma-international.org/12.0/#sec-overview

Latorre, M. (2018). Historia de las web, 1.0, 2.0, 3.0 y 4.0. *Universidad Marcelino Champagnat*.

LibHunt. (2022). *pyodide VS pyscript*. https://www.libhunt.com/compare-pyodide-vs-pyscript

LibHunt. (2022). *Python WebAssembly*. https://www.libhunt.com/l/python/topic/webassembly

Microsoft. (2022). *Choose Between Traditional Web Apps and Single Page Apps (SPAs)*. https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/choose-between-traditional-web-and-single-page-apps

Molina Ríos, J. R., Zea Ordóñez, M. P., Contento Segarra, M. J., & García Zerda, F. G. (2018). Comparación de metodologías en aplicaciones web. *3C Tecnología: glosas de innovación aplicadas a la pyme*, *7*(1), 1-19. https://dialnet.unirioja.es/descarga/articulo/6415697.pdf

Mozilla Corporation. (2020). *Compatibilidad con navegadores*. https://developer.mozilla.org/es/docs/WebAssembly#browser_compatibility

Mozilla Corporation. (2021). *Estructura web y documentación*. https://developer.mozilla.org/es/docs/Learn/HTML/Introduction_to_HTML/Document_and_website_structure.

Mozilla Corporation. (2021). *SPA (Single-page application)*. https://developer.mozilla.org/en-US/docs/Glossary/SPA

Mozilla Corporation. (2022). *Understanding client-side JavaScript frameworks*. https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks#introductory_guides

Potter, J. (2022). *angular vs react vs vue | npm trends*. https://www.npmtrends.com/angular-vs-react-vs-vue

Pyodide. (2021). *pyodide/pyodide* (0.20.0). https://doi.org/10.5281/zenodo.5156931

PyScript. (2022). *PyScript | Run Python in Your HTML*. https://pyscript.net

Speedtest. (2022). *Bolivia's Mobile and Fixed Broadband Internet Speeds*. https://www.speedtest.net/global-index/bolivia

Stack Overflow. (2022). *Stack Overflow Trends*. https://insights.stackoverflow.com/trends?tags=reactjs%2Cvue.js%2Cangular

WebAssembly. (2020). *WebAssembly*. https://webassembly.org/

World Wide Web Consortium. (2016). *HTML & CSS*. https://www.w3.org/standards/webdesign/htmlcss

World Wide Web Consortium. (2016). *JAVASCRIPT WEB APIS*. https://www.w3.org/standards/webdesign/script

World Wide Web Consortium. (2021). *ABOUT W3C*. https://www.w3.org/Consortium/

World Wide Web Consortium. (2021). *STANDARDS*. https://www.w3.org/standards/