**Ubuntu Linux como AppServer**

**Zini, Horacio Antonio**

Fac. Ciencias Exactas, UNNE, Corrientes, Argentina

**1 Instrumentación**

Para la construcción del servidor de aplicaciones se utilizaron las siguientes herramientas:

* Python
* Flask
* Socket.IO
* Servidor WSGI de Eventlet
* Oracle VM Virtual box
* Sistema operativo linux ubuntu.

De igual manera que en la sección anterior el sistema operativo estará montado sobre Oracle VM. Lo primero que tendremos que hacer es instalar la aplicación Python introduciendo los siguientes comandos:

$sudo apt-get update

$sudo apt-get install python3

Lograda la instalación de Python ahora trabajaremos con **pip**, es un sistema de gestión de paquetes utilizado para instalar y administrar paquetes de software escritos en Python. Los comandos que introduciremos serán:

$pip install Flask==1.1.2

$pip install Flask-Session==0.3.2

$pip install Flask-SocketIO==4.3.1

$pip install python-engineio==3.13.2

$pip install python-socketio==4.6.0

$pip install Werkzeug==2.1.0 --force-reinstall

$pip install eventlet

\* Las versiones usadas pueden variar en cada caso, pero esta es una combinación válida de versiones para evitar problemas de compatibilidad.

Con esto estamos logrando la instalación de lo siguiente:

* **Flask:** es un framework web, módulo de Python ideado para desarrollar aplicaciones web.
* **Socket.IO:** es una biblioteca de Python que permite comunicación en bidireccional en tiempo real, basada en eventos, entre los navegadores clientes y el servidor.
* **Flask-SocketIO:** es un paquete que permite la integración de Socket.IO con aplicaciones Flask.
* **Eventlet:** es una biblioteca de redes escrita en Python. Logra una alta escalabilidad y concurrencia mediante el uso de io no-bloqueante y, al mismo tiempo, conserva una alta usabilidad del programador mediante el uso de corrutinas para hacer que las operaciones de io sin bloqueo parezcan bloquearse en el nivel del código fuente.
* **servidor WSGI:** de Eventlet proporciona una manera simple y fácil de iniciar un servidor WSGI controlado por eventos. Esto puede servir como un servidor web integrado en una aplicación o como base para un paquete de servidor web más completo.

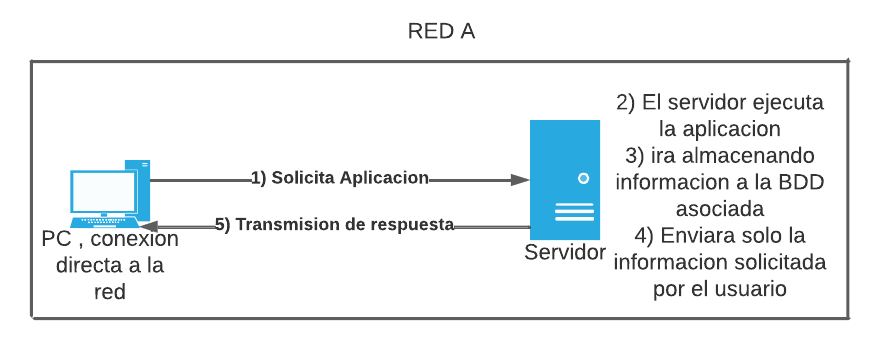
**2 Aplicación Web Cliente-Servidor de Chat para redes de área local**

Un servicio es un recurso que expone a una aplicación a tráfico externo.

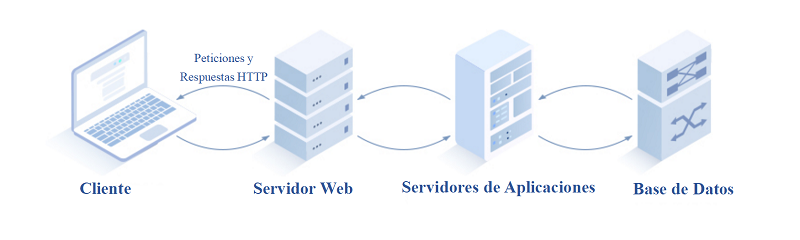
Un servidor es una máquina, física o virtual, integrada a una red, que ofrece un servicio especial que otros programas denominados clientes pueden usar a nivel local o a través de internet. El tipo de servicio depende del tipo de software del servidor. La base de la comunicación es el modelo cliente-servidor y, en lo que concierne al intercambio de datos, entran en acción los protocolos de transmisión específicos del servicio.

Una denominación alternativa para un servidor basado en hardware es "host" o anfitrión. En principio, todo ordenador puede usarse como "host" con el correspondiente software para servidores.

Dejando esto en claro es momento de establecer un concepto de **servidor de aplicaciones**. Los servidores de aplicaciones son programas de servidor en una red distribuida que proporciona el entorno de ejecución para un programa de aplicación. Más específicamente, el servidor de aplicaciones es el componente de tiempo de ejecución principal en todas las configuraciones y donde una aplicación se ejecuta realmente. Un esquema básico se aprecia en la siguiente figura:



En la actualidad por lo general el servidor de aplicaciones **colabora** con el servidor web para ofrecer una respuesta dinámica y personalizada a una solicitud de cliente. De forma que el esquema se representa de la siguiente forma:



En el desarrollo del aplicativo esta colaboración se aprecia mediante el desarrollo de la aplicación utilizando Python , que es ejecutada en la máquina virtual, y la prestación del servidor web WSGI.

La Interfaz de Puerta de Enlace del Servidor Web, "WSGI" (Web Server Gateway Interface), es una convención de llamada simple para que los servidores web re-envíen solicitudes a aplicaciones web o marcos escritos en el lenguaje de programación Python.

Eventlet es una biblioteca de redes escrita en Python que, entre otras cosas, ofrece un servidor WSGI el cual proporciona una manera simple y fácil de iniciar un servidor controlado por eventos. Esto puede servir como un servidor web integrado en una aplicación o como base para un paquete de servidor web más completo. Logra una alta escalabilidad y concurrencia mediante el uso de IO no-bloqueante y, al mismo tiempo, conserva una alta usabilidad del programador mediante el uso de corrutinas.

Una vez instalados los recursos especificados en la sección 2.2, el procedimiento para implementar la aplicación consiste de los siguientes pasos:

1. Crear un directorio, preferentemente en el directorio “home”, que será la base del proyecto.
2. En dicho directorio, se crearán los archivos “app.py” y “app.wsgi” con el código especificado en la sección siguiente.
3. Se creará también la carpeta la carpeta “templates” que contendrá las dos plantillas o interfaces que usará nuestra aplicación
4. En “templates” irán los archivos “index.html” y “chat.html” con el código especificado debajo.

**2.1 Ejecución de la aplicación**

Una vez completados los pasos anteriores, para ejecutar la aplicación basta con definir la variable de ambiente FLASK\_APP que contendrá el camino absoluto al archivo .py que es la aplicación

FLASK\_APP = "/home/…/app/app.py"  
o bien, posicionarse, con el comando cd, en el directorio que contenga el archivo, y ejecutar el siguiente comando:

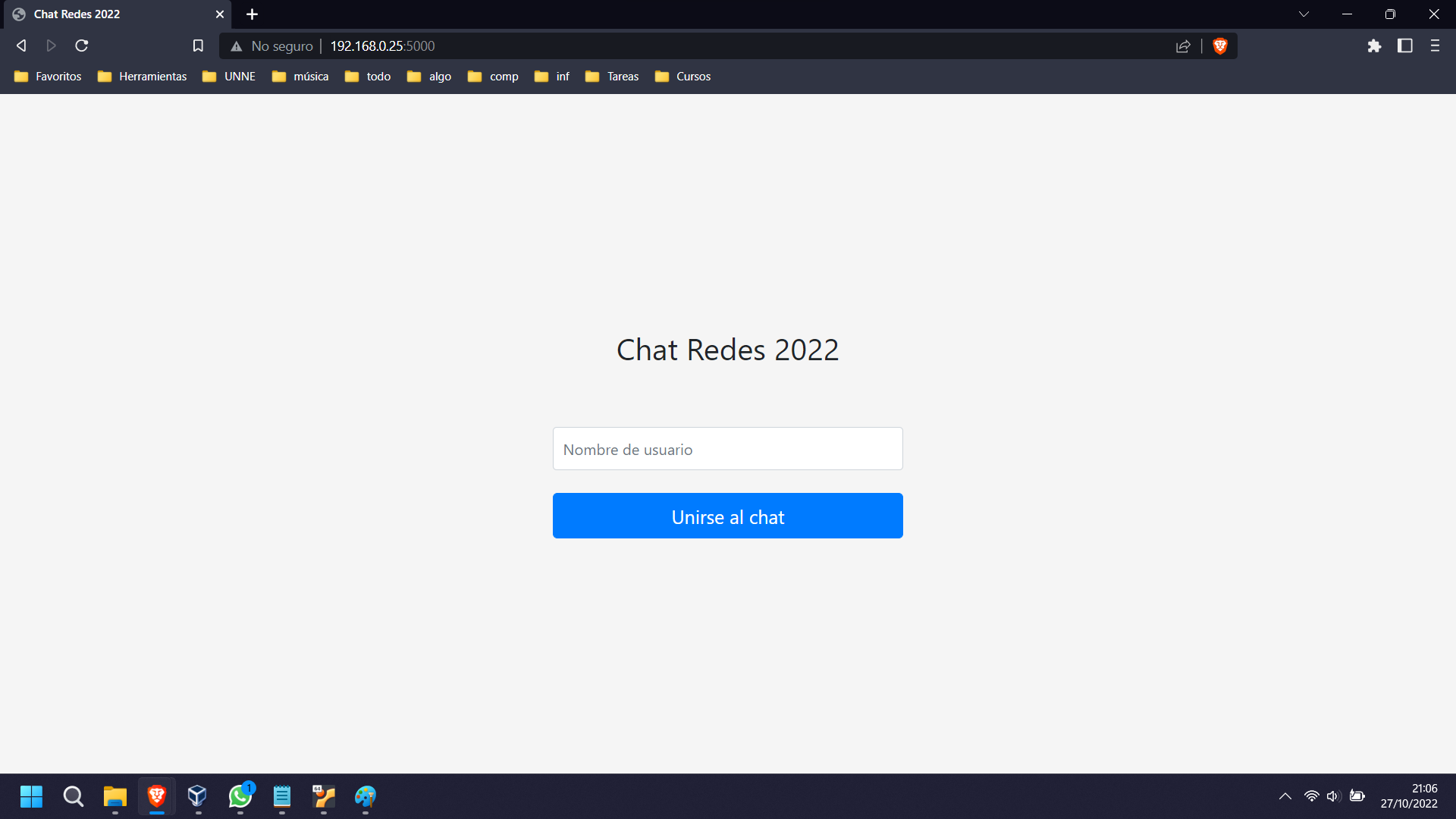
sudo flask run -h <IP local>

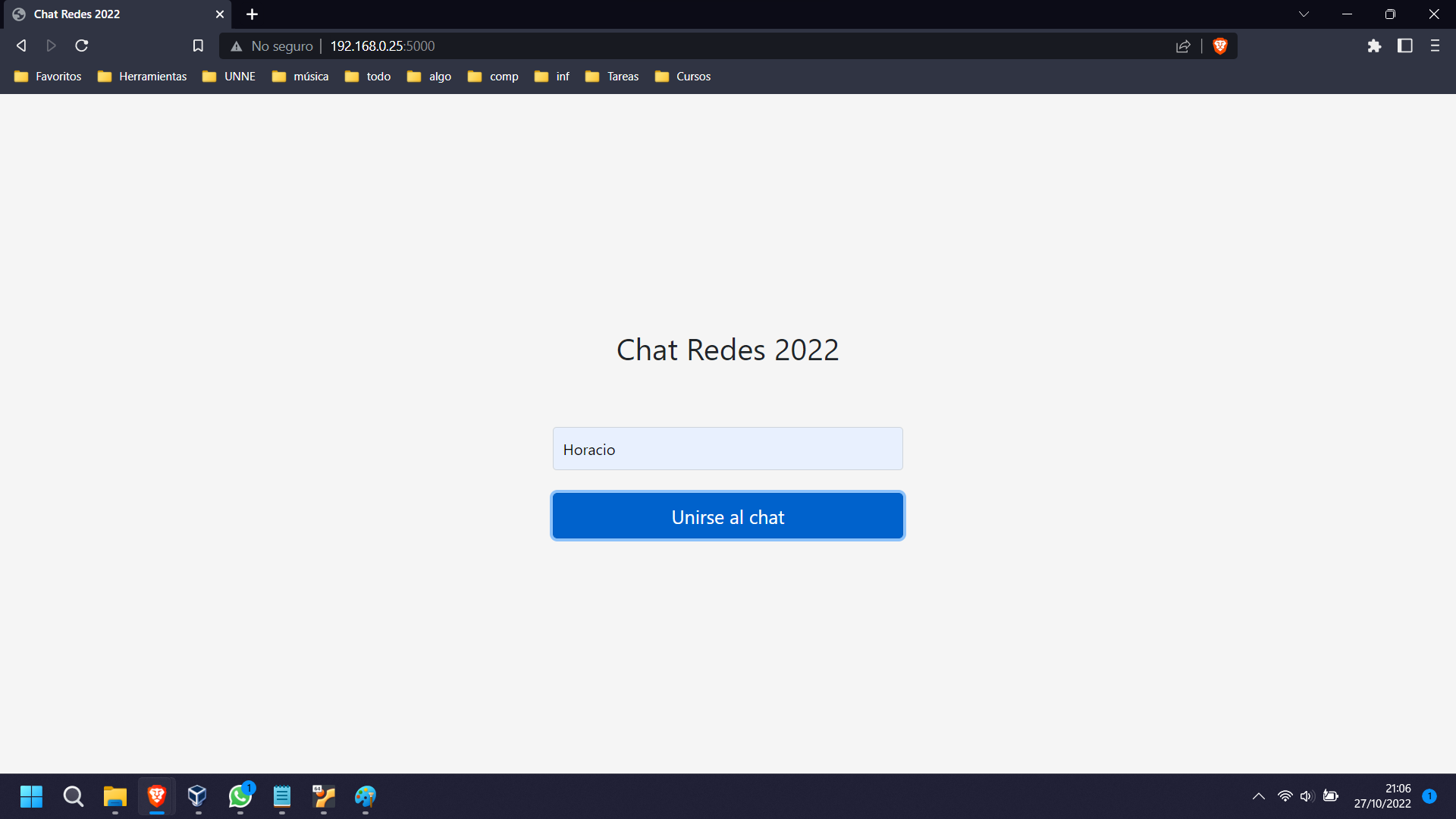
Por ej: sudo flask run -h 192.168.0.25

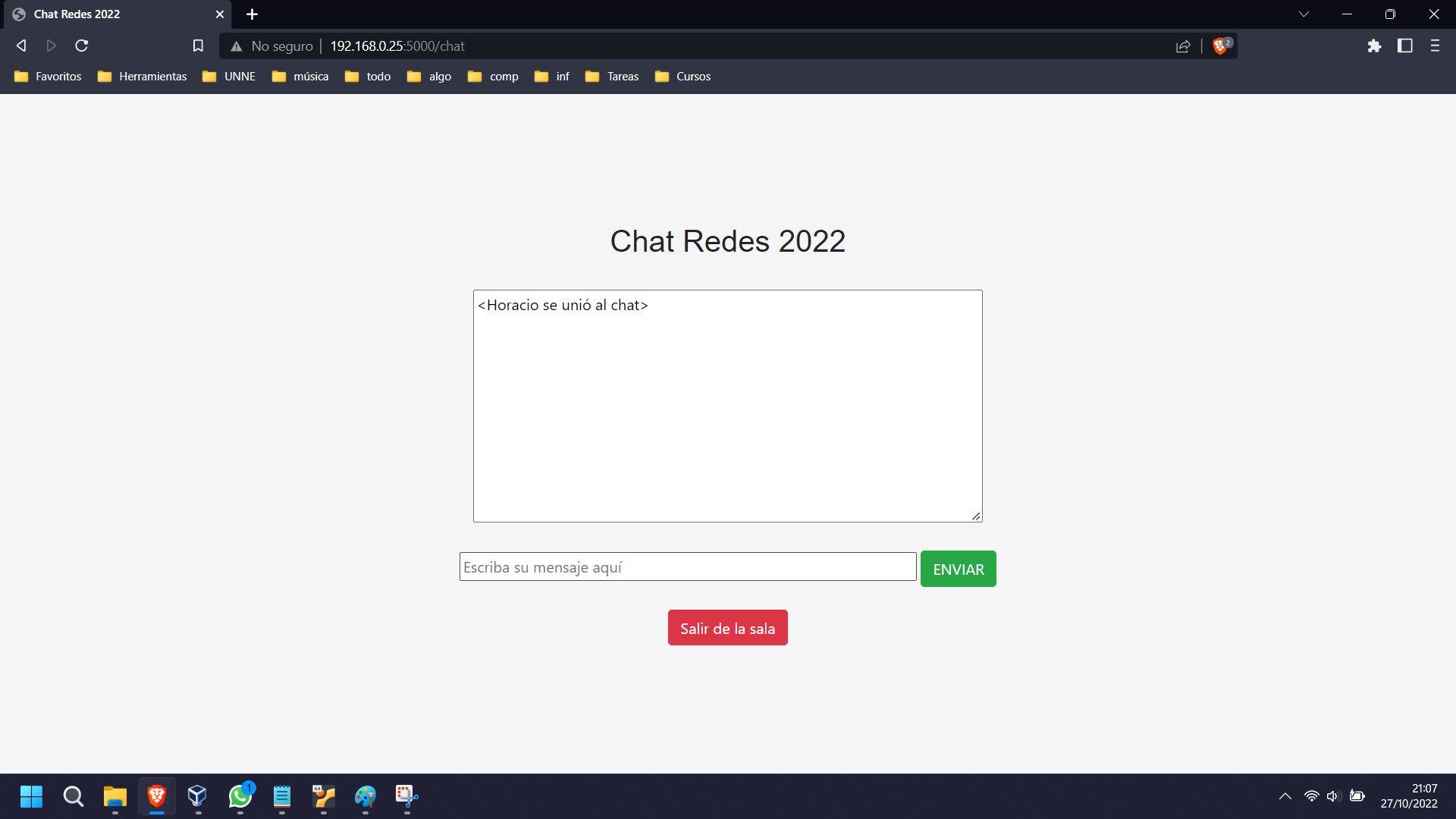
Que devolverá la linea “Running on http://192.168.0.25:5000” que nos indica que la aplicación está siendo servida exitosamente en el puerto 5000 del local host.

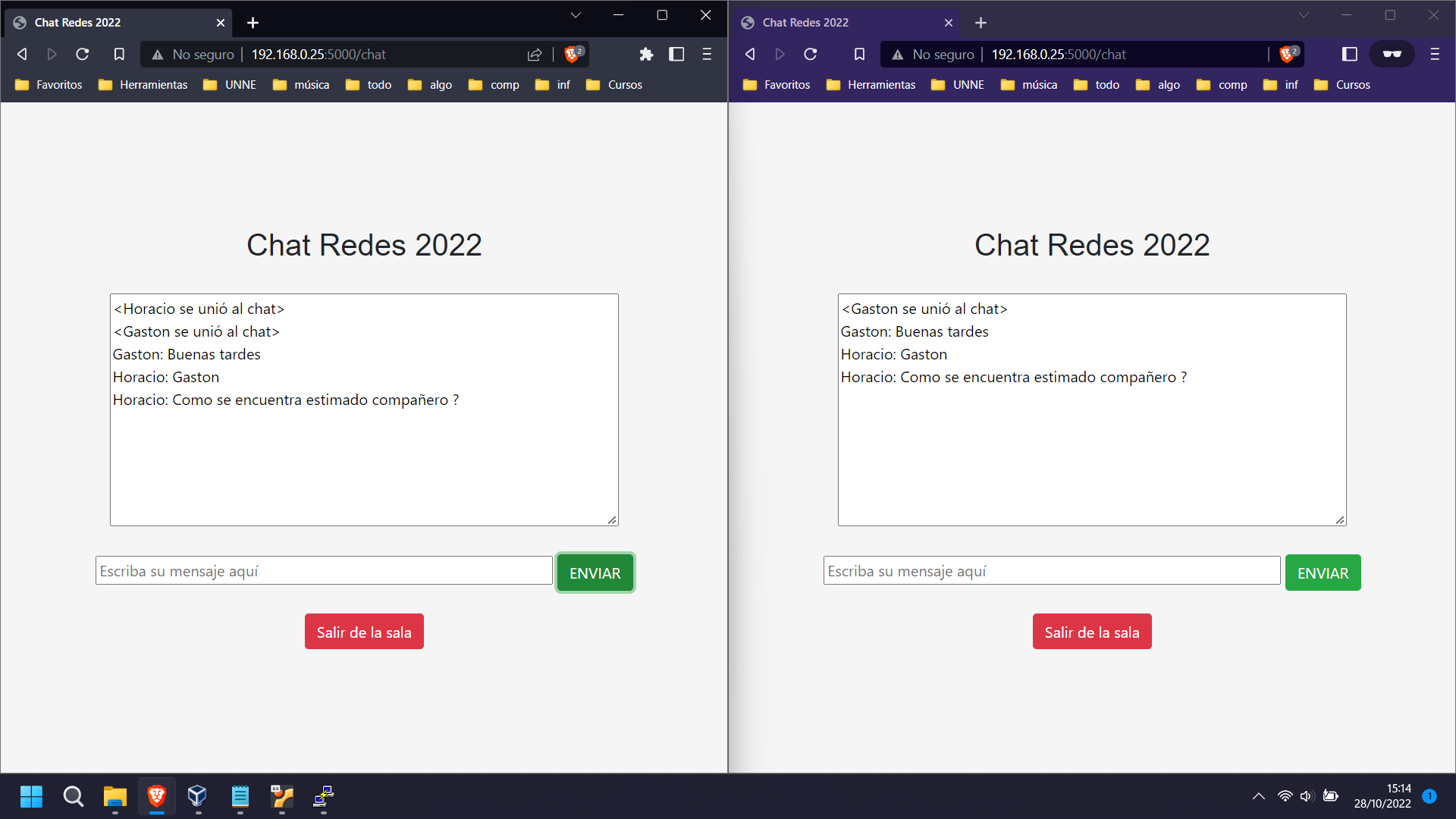
A partir de este momento y hasta que se dé de baja el servidor, es posible acceder a la aplicación desde los navegadores de cualquier dispositivo conectado a la misma red local que la máquina que ejecuta el servidor, ingresando IP:PUERTO en la barra de dirección.

**2.3 Resultados de la ejecución**

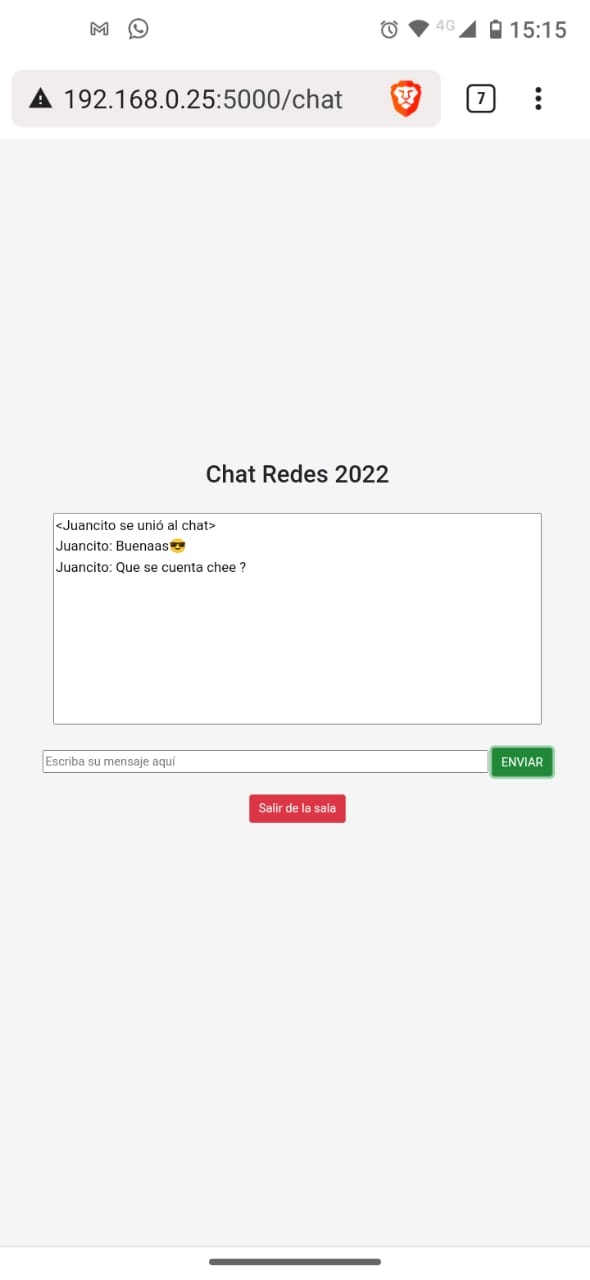


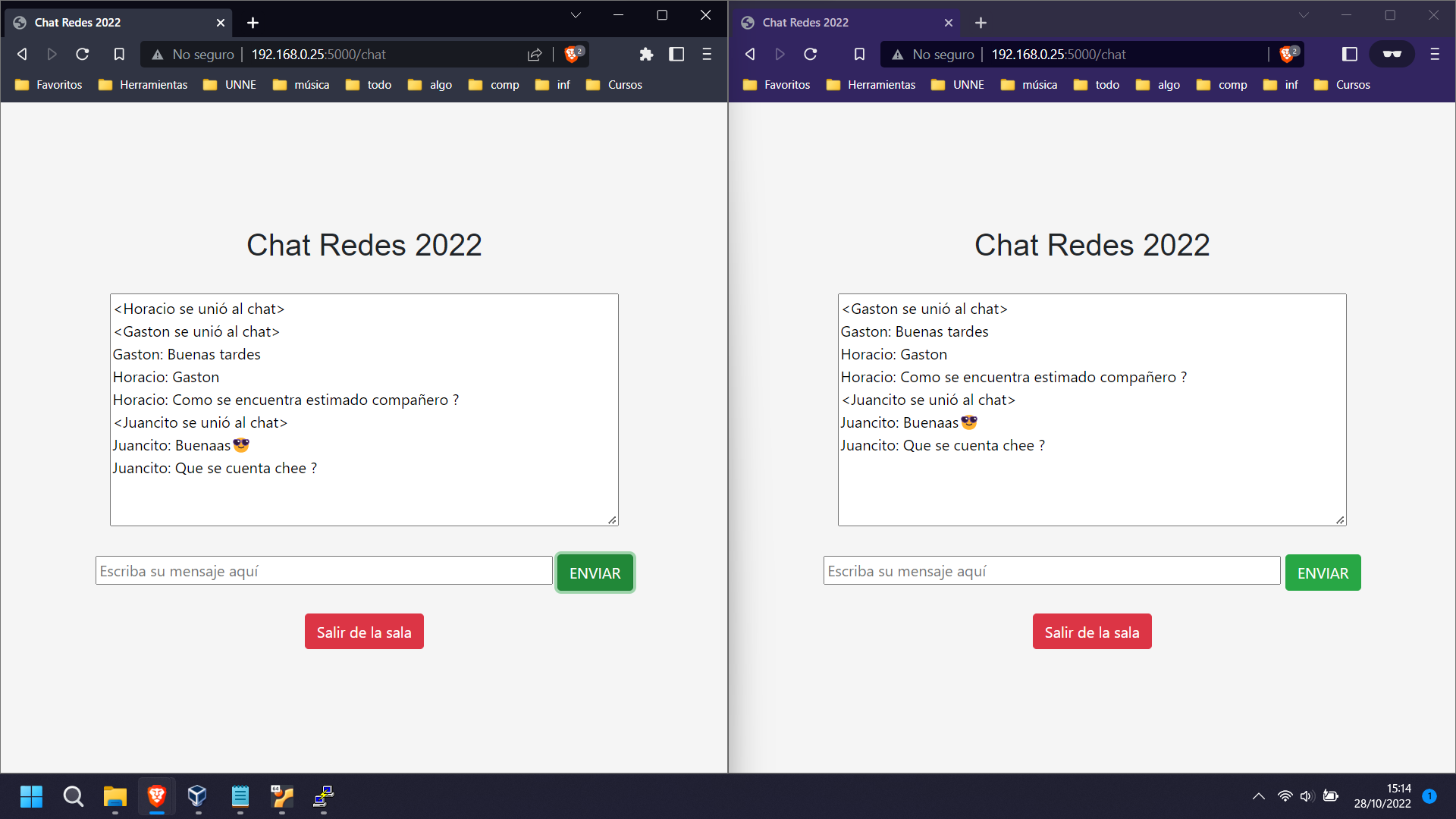






Ingresando desde Android:





**2.4 Ventajas y desventajas del servidor de aplicaciones**

**Ventajas:**

**Ahorro de recursos para el cliente**: El cliente que consume las aplicaciones ofrecidas no tiene necesidad de presentar un equipo de ‘alto nivel’ para su uso. Así como no necesita tener instalado los intérpretes de los lenguajes que utiliza la aplicación.

**Aplicación en web:** al implementar este servicio mediante un servidor web tenemos la posibilidad de desarrollar la aplicación a un mayor nivel permitiendo así clientes de forma remota que se conecte mediante algún servicio DNS.

**Adaptabilidad:** Otra ventaja de poseer una aplicación web es que la misma se adapta a los distintos dispositivos. Este caso se aprecia en la sección 4.3 donde se logró un acceso mediante un sistema operativo Android desde un dispositivo móvil.

**Desventajas:**

**Complejidad de despliegue:** Una vez desarrollada la aplicación, es muy importante conocer de las características de los lenguajes utilizados para poder decidir la mejor opción entre los servidores a utilizar para el despliegue, así como todo los recaudos necesarios.

**Mayor consumo del lado del servidor:** El peso de ejecutar aplicaciones, teniendo en cuenta la posibilidad de múltiples llamadas al servicio de manera simultánea, es una característica a considerar, ya que un servidor que no cumpla con los requerimientos sería muy susceptible a fallos y a caídas.

**Necesidad de infraestructura de redes:** Al hacer uso del servidor de aplicación mediante la web será necesario tener un equipamiento de infraestructura de redes que soporte la gran cantidad de solicitudes de información. Y también tiene que lograr una buena y rápida transmisión de la información solicitada.

**3 Conclusiones**

Podemos apreciar su utilidad realizando ciertos contrastes con un servidor web, pues ambos constituyen sistemas visibles en la red a la que están conectadas para ofrecer ciertos recursos a los clientes que accedan a él, la diferencia radical es que por servidor web se entiende a el conjunto de elementos que ofrecen un sistema construido en lenguajes de marcado como ser HTTP y CSS, todo esto apoyado por un motor web como ser un motor de planillas, escrito en algún lenguaje interpretado como PHP. Por servidor de aplicaciones, en cambio se entiende a un sistema análogo que, además, debe disponer de los elementos necesarios para permitir la publicación de contenido escrito en lenguajes de programación con un mayor poder de procesamiento, como Python, C#, Java, entre otros. Estos servidores de aplicaciones permiten una gama más alta de utilidades que los servidores web tradicionales, al mismo tiempo que tienen un mayor consumo de recursos y se necesita de un mayor manejo de paralelismo y balance de carga, tanto como consideraciones en cuanto a aspectos como la seguridad y la robustez del sistema.

**4 Bibliografía**

* https://docs.python.org/3/
* https://flask.palletsprojects.com/en/2.2.x/
* https://socket.io/docs/v4/
* https://flask-socketio.readthedocs.io/en/latest/
* http://eventlet.net/doc/
* https://eventlet.net/doc/modules/wsgi.html
* https://peps.python.org/pep-3333/