**2. Desarrolla aplicaciones compuestas por varios hilos de ejecución analizando y aplicando librerías específicas del lenguaje de programación.**

Criterios de evaluación:

**a) Se han identificado situaciones en las que resulte útil la utilización de varios hilos en un programa.**

La utilización de varios hilos en un programa es útil cuando tenemos que asignar tareas a diferentes usuarios para que las procesen y devuelvan estos datos al servidor. En este caso y como se verá en la práctica que se detalla en el video, hicimos una pequeña aplicación en la que los usuarios debían identificarse y el servidor, cuando ya estaban identificados les asignaba una tarea. Esta tarea consistía en un array de números que cada usuario debía sumar y luego devolver el resultado al servidor. Para ello usamos html y php. Además, para guardar los datos y que no se perdieran los usuarios registrados se utilizaron archivos txt donde se registraban los nombres de los usuarios que iban identificándose. También se creo un archivo txt con las tareas existentes que iban eliminándose conforme se asignaban a los usuarios. Estas asignaciones se recogían en otro archivo para saber a que usuario le había tocado cada tarea. Por último, para tener constancia de que los usuarios estaban procesando las tareas y devolviendo los datos, en otro archivo se registraban los resultados devueltos.

Además, para hacer más visual el ejercicio, personalmente he añadido una interfaz de usuario mas intuitiva y dinámica donde el usuario puede saber si se le ha asignado correctamente la tarea o si ha ocurrido algún error También hemos puesto parámetros para evitar que se repitan los usuarios o que se envíen registros de usuario sin nombre.

Por otro lado, he creado una interfaz para dar más visualización a la asignación de tareas y los resultados generados que son devueltos al servidor. En un archivo html he creado una tabla donde se muestran en pantalla esos datos. Así podemos ver de manera rápida y clara los usuarios que hay y los resultados que cada uno de ello devuelve.

**e) Se han utilizado mecanismos para compartir información entre varios hilos de un mismo proceso.**

En este caso, hemos creado un chat para enviar y recibir mensajes entre varios usuario que se conecten a una misma IP. Lo hemos hecho usando Python flask que nos permite crear un servidor con estados a diferencia de php con el que solo podemos crear un servidor stateless. Un servidor stateless con php se caracteriza porque al arrancarlo devuelve algo pero una vez que lo ha hecho se detiene. Y si vuelves a reiniciarlo, todo vuelve a empezar de nuevo. En cambio, un servidor con estados si está constantemente escuchando, está todo el tiempo ejecutándose y eso es lo que nos permite mantener la información sin que se pierda.

En esta aplicación, lo que hemos hecho con Python es crear un pequeño servidor con la librería flask que nos va a permitir crear un chat con el que enviar mensajes. Es un servidor muy sencillo que nos permite crear una lista mensajes que estará vacía al arrancarse. Cuando alguien llame a la ruta “dame” se crea una variable mensajes donde se meten los datos enviados y se devuelve por pantalla. Y también creamos una ruta “toma” que va a atrapar esos mensajes en la URL y lo añade a la lista mensaje.

En cuanto a la interfaz de usuario para poder enviar y recibir los mensajes, se ha creado un visor de estos mensajes donde podemos saber que usuario está enviado cada mensaje y cual es el mensaje que envía en cada caso. Además, cada usuario tendrá asignado un color para hacer más sencillo la identificación de ca uno de ellos. Aquí es interesante mencionar el uso de una función bucle que permite hacer la petición al servidor y está constantemente recibiendo los datos mediante un temporizador que se activa cada 1000 milisegundos.