Minutas

Integrantes:

Bernal, Sebastian

Ruiz, Agustin

Semana 04/10

- Se comenzó a plantear el cliente- servidor con una interfaz pura en html y utilizando HTTP vanilla de node.
- Se definieron nombres para las variables de modo de lograr consistencia.

Semana 11/10

- Se decidió utilizar express para el manejo de HTTP en el servidor.
- Se decidió utilizar css para el frontend complementando el html.
- Se decidió utilizar fetch para enviar las solicitudes POST de http en vez de utilizar el propio formulario.

Semana 18/10

- Se continuó con el desarrollo del cliente-servidor.
- Se comenzó con el desarrollo de los nodos tracker.
- Se decidió implementar la DHT como un TDA en un archivo separado.
- Se decidió persistir la DHT para que si se caía un nodo no se perdiera la información.
- Se agregó un archivo de estilos para manejar las animaciones del cliente.
- Se decidió inicializar los nodos tracker desde una configuración en json de forma estática, instanciandolos ya con sus nodos vecinos cargados.

Semana 25/10

- Se decidió cancelar la persistencia de la DHT ya que los nodos vecinos al nodo que se cae tendrán su información de forma redundante.
- Se decidió que las interfaces tracker-tracker no tenían la ruta por UDP sino que iría dentro del mensaje como identificador de la operación.
- Se decidió definir un rango de valores fijo para cada DHT de los trackers.
- Se decidió que el hash de un archivo lo hace el servidor y se lo manda al tracker.
- Se decidió que el dominio de la DHT de cada tracker será calculado dividiendo 127 entre la cantidad de nodos tracker y luego utilizando el ID de cada uno para definir el inicio y el fin del mismo.
- Se decidió chequear por condición el tipo de ruta de cada request (ej: /store).
- Se estableció que el formulario de carga contiene: nombre de archivo, tamaño en bytes y dirección del nodo par donde se encuentra.
- Se realizó la interfaz Search y Found desde el lado del Tracker.
- Se cambiaron nombres de clases de elementos para manejar los estilos de forma más personalizada.
- Se realizaron cambios de animaciones principales y colores.

Semana 1/11

- Se descartó tener un formulario de descarga.
- Se definió una función en cliente.js para generar contenido HTML desde javascript. Pudiendo armar una lista de descargas dinámicamente con los archivos disponibles.
- Se decidió que cada elemento disponible para descargar tenga su propio botón en vez de ser un link.
- Se agregaron timeouts para tener sincronismo entre la carga del mensaje y el envío desde el server al tracker.

Semana 28/11

- Se decidió crear labels para la lista de archivos a descargar.
- Se decidió usar ‘fetch’ para manejar la solicitud de descarga.

Semana 5/12

- Se hicieron pruebas para lograr recibir la lista de archivos disponibles al Cliente.
- Se usaron links para cada archivo disponible de la lista.
- Se decidió reemplazar los links por botones individuales para cada archivo de modo de acceder a los eventos.
- Se decidió tener una función capaz de mapear cada botón presionado de la lista con su correspondiente índice del array de archivos.
- Se cambió el contenido de los archivos .torrente para que contengan la información de la dirección del tracker asociado.
- Se realizó la estructura de los nodos Par.
- Se realizaron correcciones en la interfaz Search y Found desde el lado del Tracker.
- Se definió la forma de descargar los archivos .torrente.
- Se establecieron timeouts para lograr sincronismo entre la carga de mensajes y el envío.
- Se estableció la posibilidad de consultar al nodo Par por consola.
- Se definió un switch para manejar el input por consola del nodo Par.
- Se definió un archivo de configuración para los nodos Par.
- Se decidieron distintos inputs para el nodo Par: “exit”, “id”, “descargar” y “nombre de archivo”.
- Se definió que la lista de archivos mostrada por el Cliente tenga nombre y tamaño de archivo así como también el botón de descarga propio de cada uno.
- Se definió una función solicitudDescarga() para descargar contenido de un Par a otro.

- Se realizaron testeos de envío de archivos de distintos tipos entre nodos Par.
- Se decidió utilizar un array de “chunks” para luego almacenar el archivo de forma local en el Par que lo solicitó.
- Se definió una interfaz avisaTracker().
- Se corrigió un bug que no dejaba subir un archivo correctamente.
- Se desinstalaron algunos paquetes que no fueron utilizados.
- Se definió la interfaz Count entre nodos Tracker.
- Se mejoró la subida de archivos.
- Se hicieron cambios en las animaciones y layout de Lista de descargas.