* camlink/start/
  + Method: POST
  + Send: none
  + Response: {‘msg’:’success’, ‘code’:’<the link code>’} and a cookie
  + Backend:
    - Set code variable to the link code before returning it to the client
    - This code will serve as entry point when clients want to join
    - Peers:0 in the cookie
* camlink/link/
  + Method:POST
  + Send:{‘code’:<link code>}
  + Response: {‘msg’:’success’}
  + Cookie:<session id>
  + Backend:
    - Receive the code, check if it matches with the code variable
    - Create a new user in the camlink table
    - Create a new session and save the id of the user in it
    - In the videos folder create a new folder by the name of the user id. This folder will store the chunks of videos which are suitable for streaming.
    - And in that folder create another folder named full. This folder will store a single video by the name of the user id again, but this single video is the whole video resulted from the chunks of videos, suitable for playing it and watching.
    - Send the session id as a cookie to the client
* camlink/stream/
  + Method:POST
  + Send:{‘chunk’:<video chunk>}[formdata] and cookie
  + Response: {‘msg’:’success’}
  + Backend:
    - Receive the chunk of video and get the user id using the session id in the cookie
    - And then go to the folder by the name of the user id and save the chunk of video. The naming of the chunks should start from 0 and continue in ascending order. The number that the user is currently on is available in the camlink table.
    - Add the number of the user by one
    - Go to the video inside the full folder and append the video chunk
* camlink/refresh/
  + Method:GET
  + Send:none
  + Response: {‘msg’:’success’,’add’:[<list of new user ids>]}
  + Response: {‘msg’:”no new link”} if there is no new link
  + Backend:
    - Check if the peers is less than the number of users in camlink table.
    - If so then send all users (their ids) above the peers number in the cookie to the client and update peers to the latest user id.
* camlink/play/
  + Method:GET
  + Send:{“uid”:<user id>,”cid”:<chunk id>}
  + Response: the requested chunk of video
  + Backend:
    - Return the requested chunk of video
* camlink/stop/
  + Method:POST
  + Send:none
  + Response: {‘msg’:”success”}
  + Backend:
    - Clean the Link database
    - Export the videos from each camera to “exports” folder
    - Delete all folders in the videos folder