Desktop Streamer

# Introduction

Goal is to steam desktop screen and control remotely the mouse and keyboard at high framerate (60 fps) and resolution (1920x1080 - 12Mbps) in low latency (under 100ms).

## Components of the program

User (Host)

Special browser (electron)

JavaScript

Server

nodejs

SQLite

WebRTC

User (Peer)

Browser

JavaScript

## Usage flow

Just like a webcam chat application but user can control mouse and keyboard.

Connect to server

(only in special browser)

Login page

(skip if guest)

Register

Password recovery

Settings

Main page

(skip if guest)

add room

remove room

share room

enter room

Room

if in special browser

View the shared screen and control mouse + keyboard

Host the screen and mouse + keyboard

## Database

verifications:

verification\_id

email

code\_email

code\_local

expire

delete:

user\_id

expire

admins:

user\_id

users:

user\_id

email

username

password

sessions:

session\_id

user\_id

last\_login

last\_ip

expire

is\_recovery

recovery\_email

resources:

resource\_id

parent\_resource\_id

name

is\_room

is\_open

permissions:

permission\_id

resource\_id

user\_id

permission

# Setup

## Server

* setup dependencies: “npm install”
* edit configuration file (default path conf/conf.json)
  + port: port where can access to the server
  + key/cert: path to the key and certificate file (enable HTTPS) (optional)
  + db: path to the db file (if not exit it will create a file)
  + users: array of user object (“email”, “name”, “password”) that need to create when create the database
  + email: object that contains SMTP email setup and account features: password recovery, register (optional)
    - smtp: host, port, user, pass, from and name of the administrator email account
    - allowRegister: anybody can register an account
    - allowSameEmail: allow to reuse same email but for login will accept only username
* run server: “node server.js --configuration <conf.json path>” (default path conf/conf.json)

## Special browser

Application can customized in “resources/app/conf.json”. The server access can locked in “server” key that contains address of the server.

# Usage

## Account security

User can create account with configuration or registration (if enabled)

Password recovery only works if user use email and smtp server added (only initial user could be without email)

Any email or password changes generate 1 undeletable session key and send to the user’s email address. This available for 24 hours and if new one until 1 undeletable session is available.

## Room usage

In main page user can list own and shared rooms (can filter it) or join in a room

Permissions:

Owner – Can add and delete users to the room

Writer – Can add and delete users to the room (but cannot delete owner)

Reader – Only can use the room, cannot invite just exit from room

### In room:

There will be list about members, same account can be multiple times (e.g. admin, admin (2), …). After selecting member user can choose which monitor, sound want to use.

In a host there will be a share screen button. If user host the screen the program appear an option menu. Host can set which resources want to share and auto start option (this will remove if user exit from the room)