```
import { TRPCError, initTRPC } from '@trpc/server';
import superjson from 'superjson';
import { Context } from './[trpc]/context';
import { RateLimiterMemory } from 'rate-limiter-flexible';

const t = initTRPC.context<Context>().create({
  transformer: superjson,
  isDev: process.env.NODE_ENV === 'development',
});

export const router = t.router;
export const mergeRouters = t.mergeRouters;

export const publicProcedure = t.procedure;

// rate limiter
const opts = {
  points: 10,
  duration: 1, // Per second
};

const rateLimiter = new RateLimiterMemory(opts);

const getFingerprint = (req: any) => {
  const forwarded = req.headers.get('x-forwarded-for');
  const ip = forwarded
```

```
    ? (typeof forwarded === 'string' ? forwarded : forwarded[0])?.split(/, /)[0]
    : req.ip;
  return ip || '127.0.0.1';
};
export const userProcedure = publicProcedure
  .use(async (opts) => {
    const ip = getFingerprint(opts.ctx.req);
    try {
      await rateLimiter.consume(ip);
    } catch (e) {
      throw new TRPCError({
        code: 'TOO_MANY_REQUESTS',
        message: 'Too many requests',
      });
    }
    return opts.next();
  })
  .use(async (opts) => {
    const user = opts.ctx?.session?.data?.session?.user;
    if (!user) {
      throw new TRPCError({
        code: 'FORBIDDEN',
        message: "You don't have access to this resource",
      });
    }
```

```
  return opts.next();
});
```