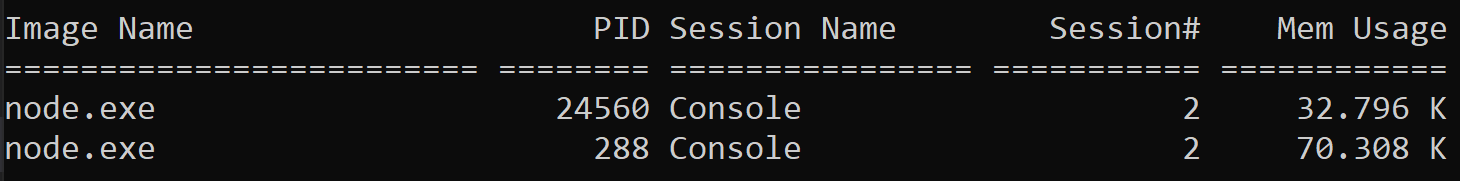
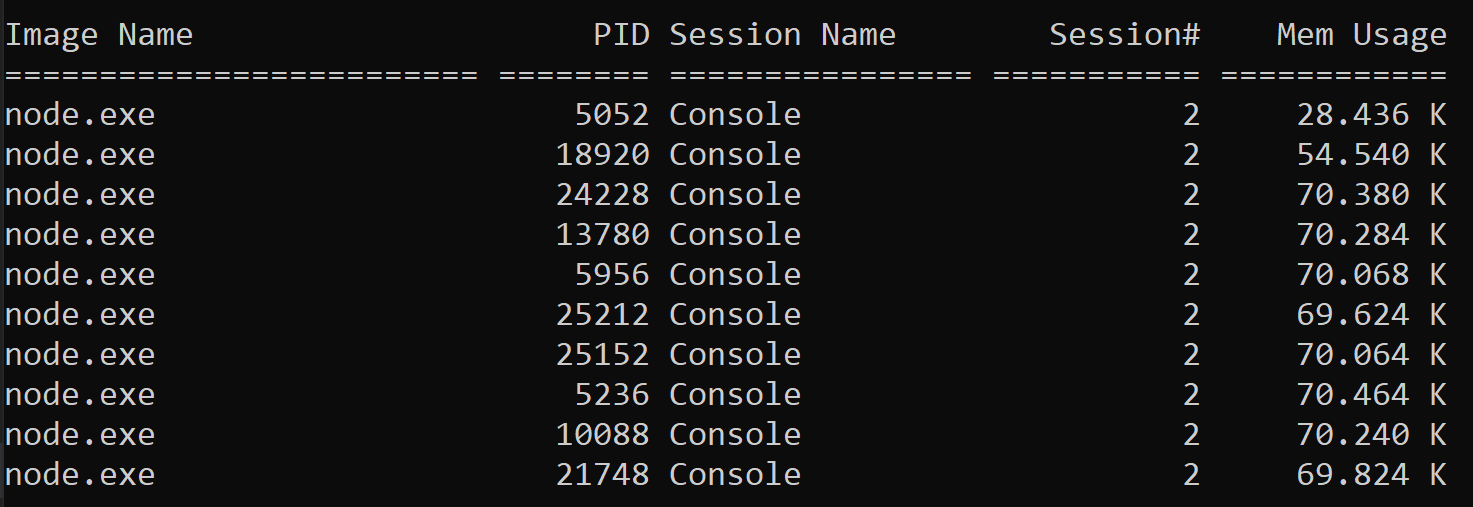
* Ejecutar el servidor (modos FORK y CLUSTER) con nodemon verificando el numero de procesos tomador por node.

Comando: nodemon index.js -p 'PORT' -m 'MODE'

Ejemplo: nodemon index.js -p 8081 -m FORK



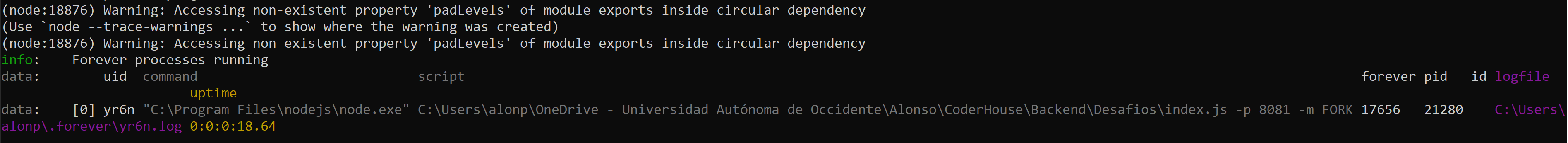
Ejemplo: nodemon index.js -p 8081 -m CLUSTER



* Ejecutar el servidor (con los parámetros adecuados) utilizando Forever, verificando su correcta operación.

Comando: forever -w start index.js -- -p 'PORT' -m 'MODE'

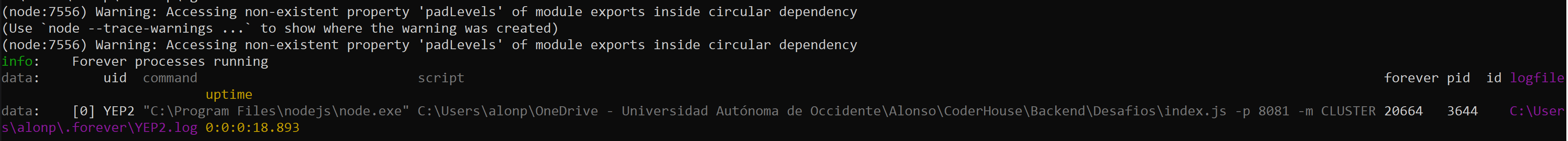
Ejemplo: forever -w start index.js -- -p 8081 -m FORK



Text

Description automatically generated with medium confidence

Ejemplo: forever -w start index.js -- -p 8081 -m CLUSTER



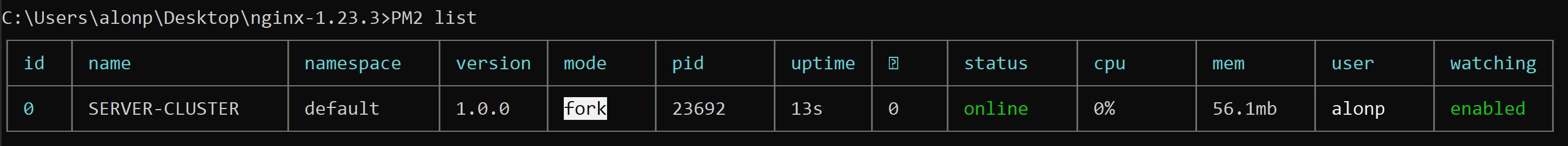
Graphical user interface, text, application

Description automatically generated

* Ejecutar el servidor (con los parámetros adecuados) utilizando PM2 en sus modos FORK y CLOSTER, verificando su correcta operación.

Comando: pm2 start index.js --watch --name 'SERVER-NAME' -- -- -p 'PORT' -m 'MODE'

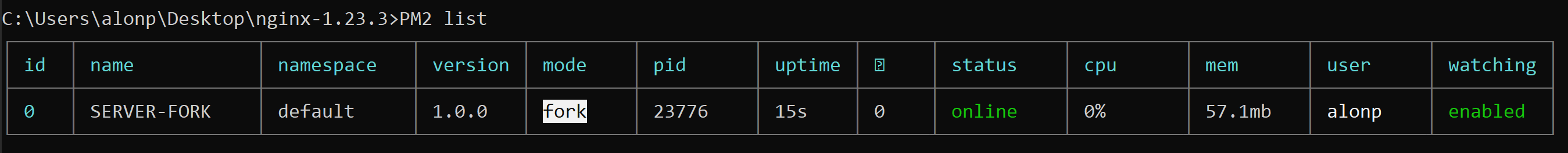
Ejemplo: pm2 start index.js --watch --name SERVER-CLUSTER -- -- -p 8081 -m CLUSTER



Graphical user interface, text

Description automatically generated

Ejemplo: pm2 start index.js --watch --name SERVER-FORK -- -- -p 8081 -m FORK



Text

Description automatically generated

* Configurar Nginx para balancear cargas de nuestro servidor de la siguiente manera: Redirigir todas las consultas a /api/randoms a un cluster de servidores escuchando en el puerto 8081. El cluster será creado desde node utilizando el módulo nativo cluster. El resto de las consultas, redirigirlas a un servidor individual escuchando en el puerto 8080.
  + Inicializar las dos instancias del servidor de la siguiente forma:
    - pm2 start index.js --watch --name SERVER-FORK -- -- -p 8080 -m FORK
    - pm2 start index.js --watch --name SERVER-CLUSTER -- -- -p 8081 -m CLUSTER
  + Archivo configuración Nginx:

worker\_processes 1;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

sendfile on;

keepalive\_timeout 65;

server {

listen 80;

server\_name proxy\_server;

location / {

proxy\_pass http://localhost:8080;

}

location /api/randoms {

proxy\_pass http://localhost:8081;

}

}

}

* Luego, modificar la configuración para que todas las consultas a /api/randoms sean redirigidas a un cluster de servidores gestionado desde nginx, repartiéndolas equitativamente entre 4 instancias escuchando en los puertos 8082, 8083, 8084 y 8085 respectivamente.
  + Inicializar las dos instancias del servidor de la siguiente forma:
    - pm2 start index.js --watch --name SERVER-FORK -- -- -p 8080 -m FORK
    - pm2 start index.js --watch --name SERVER-CLUSTER-1 -- -- -p 8082 -m FORK
    - pm2 start index.js --watch --name SERVER-CLUSTER-2 -- -- -p 8083 -m FORK
    - pm2 start index.js --watch --name SERVER-CLUSTER-3 -- -- -p 8084 -m FORK
    - pm2 start index.js --watch --name SERVER-CLUSTER-4 -- -- -p 8085 -m FORK
  + Archivo configuración Nginx:

worker\_processes 1;

events {

worker\_connections 1024;

}

http {

include mime.types;

default\_type application/octet-stream;

upstream server-cluster{

server 127.0.0.1:8082;

server 127.0.0.1:8083;

server 127.0.0.1:8084;

server 127.0.0.1:8085;

}

sendfile on;

keepalive\_timeout 65;

server {

listen 80;

server\_name proxy\_server;

location / {

proxy\_pass http://localhost:8080;

}

location /api/randoms {

proxy\_pass http://server-cluster;

}

}

}