HTTP JSON API SERVER

- Aditi Vaidya

Table of Contents

Introduction

1. Time Server and HTTP json api server is created .
2. Code is verified using `learnyounode verify file_name.js`
3. Running code is implemented.
4. Server output is displayed.

Installations

Install following on to Ubuntu Terminal using commands :-

1. Node.js

   ```
   curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
   sudo apt-get update && sudo apt-get install -y nodejs
   ```
   Check version : `node --version`

2. JSON

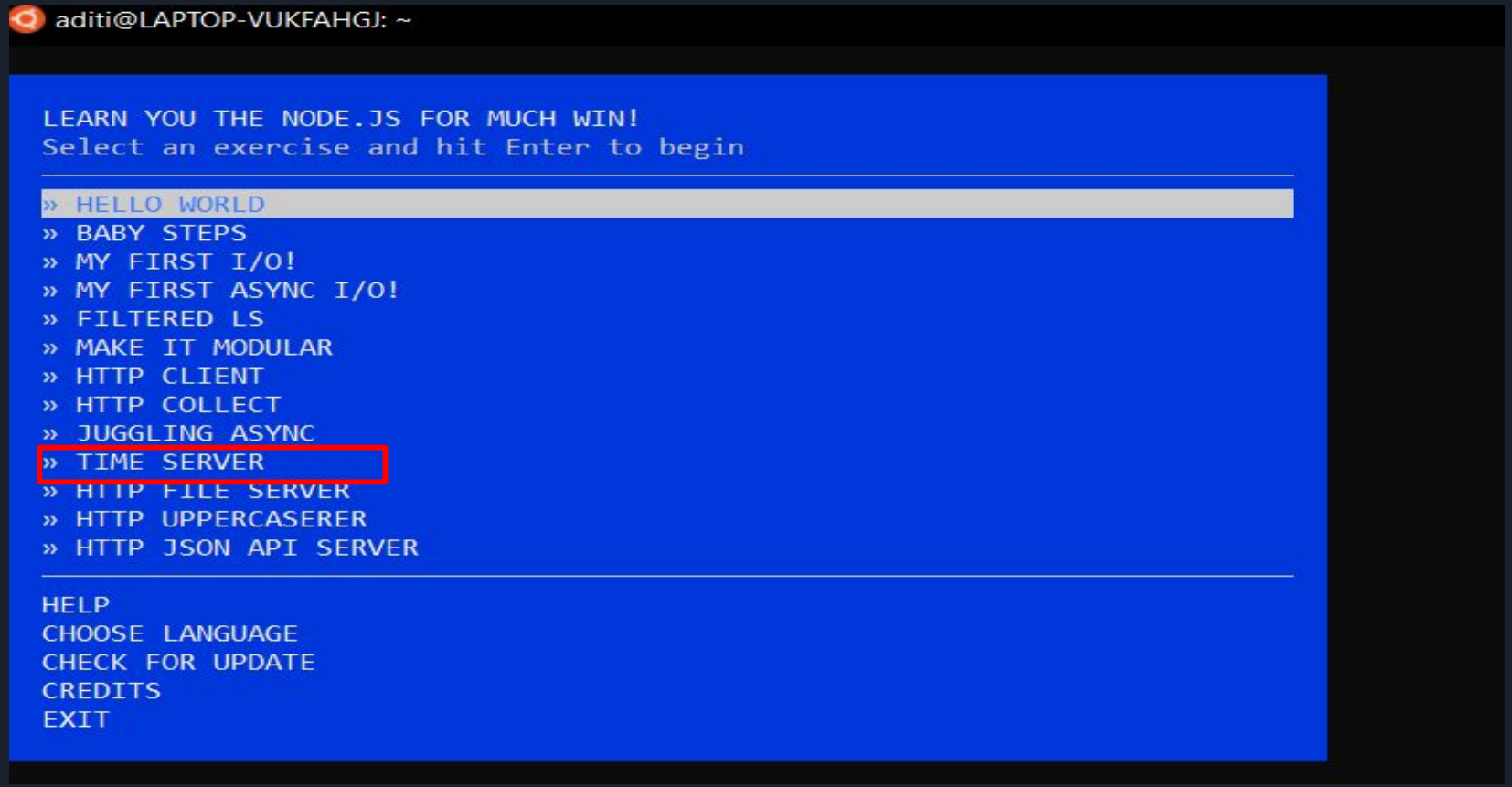
   Using `ls` check for node_modules
   `cat package.json`

   Check for dependencies.

3. NPM

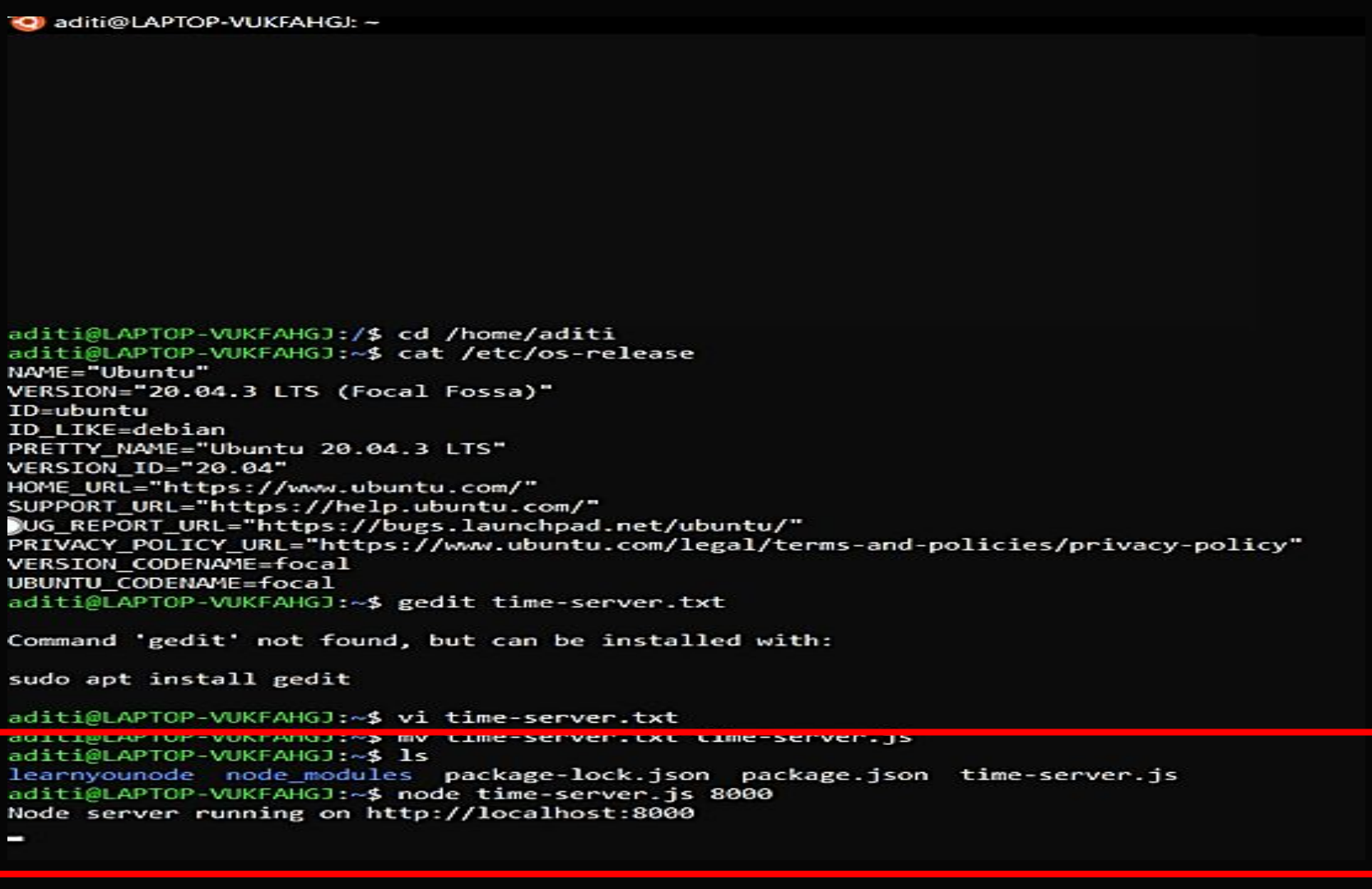
   Check version : `npm --version`

Time Server

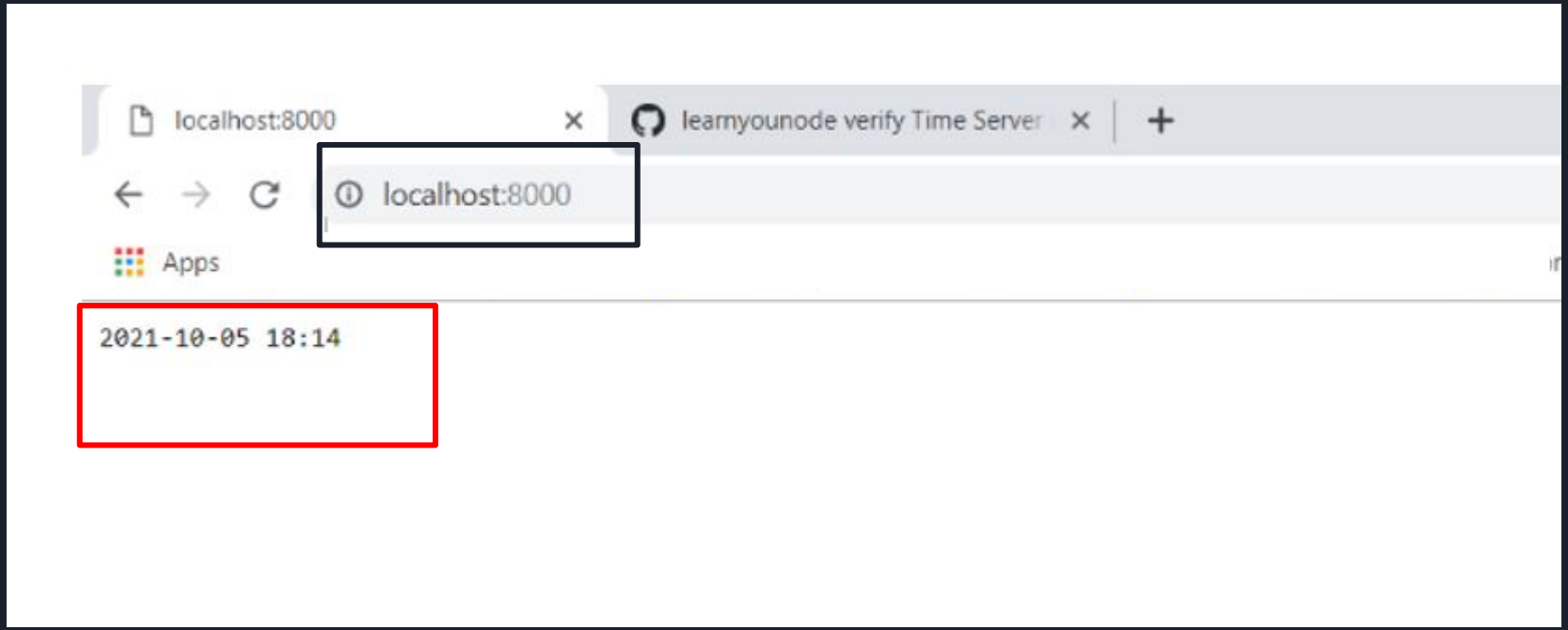

aditi@LAPTOP-VUKFAHGJ: ~
LEARN YOU THE NODE.JS FOR MUCH WIN!
Select an exercise and hit Enter to begin
» HELLO WORLD
» BABY STEPS
» MY FIRST I/O!
» MY FIRST ASYNC I/O!
» FILTERED LS
» MAKE IT MODULAR
» HTTP CLIENT
» HTTP COLLECT
» JUGGLING ASYNC
» TIME SERVER
» HTTP FILE SERVER
» HTTP UPPERCASERER
» HTTP JSON API SERVER
HELP
CHOOSE LANGUAGE
CHECK FOR UPDATE
CREDITS
EXIT

Time server running

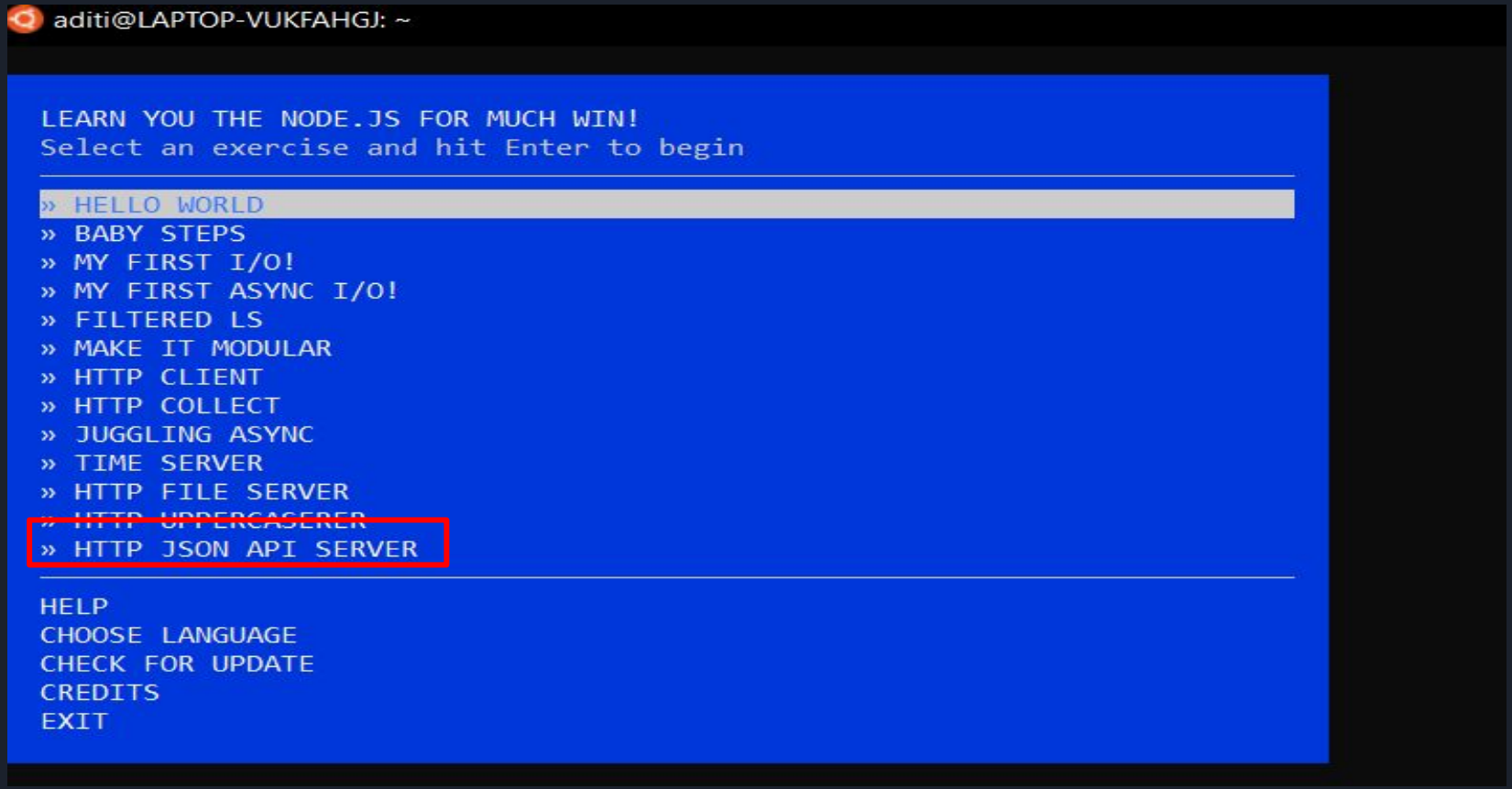
```
'use strict'
const net = require('net')

function zeroFill (i) {
  return (i < 10 ? '0' : '') + i
}

function now () {
  const d = new Date()
  return d.getFullYear() + '-' +
    zeroFill(d.getMonth() + 1) + '-' +
    zeroFill(d.getDate()) + ' ' +
    zeroFill(d.getHours()) + ':' +
    zeroFill(d.getMinutes())
}

const server = net.createServer(function (socket) {
  socket.end(now() + '\n')
})

server.listen(Number(process.argv[2]))
```


aditi@LAPTOP-VUKFAHGJ: ~
aditi@LAPTOP-VUKFAHGJ:/$ cd /home/aditi
aditi@LAPTOP-VUKFAHGJ:~$ cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.3 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.3 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
UG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
aditi@LAPTOP-VUKFAHGJ:~$ gedit time-server.txt
Command 'gedit' not found, but can be installed with:
sudo apt install gedit
aditi@LAPTOP-VUKFAHGJ:~$ vi time-server.txt
aditi@LAPTOP-VUKFAHGJ:~$ ls
learnyounode node_modules package-lock.json package.json time-server.js
aditi@LAPTOP-VUKFAHGJ:~$ node time-server.js 8000
Node server running on http://localhost:8000

Output of Time Server



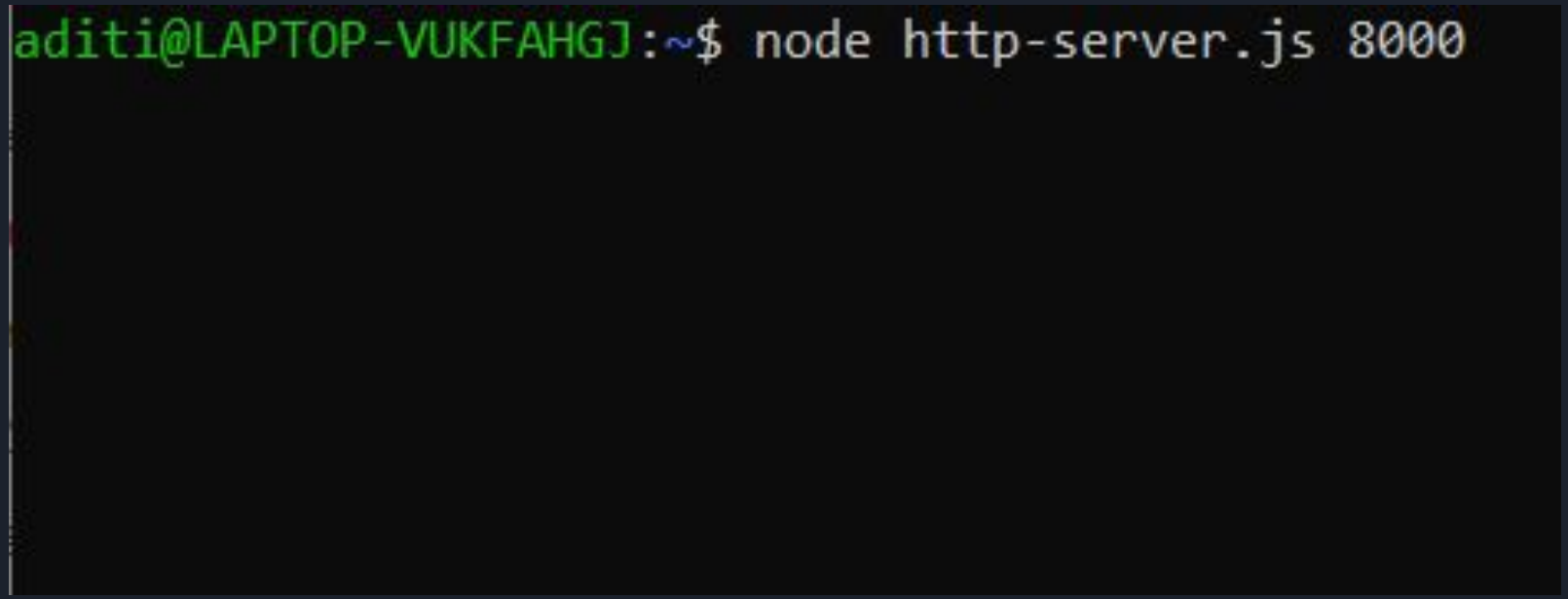
HTTP JSON API SERVER



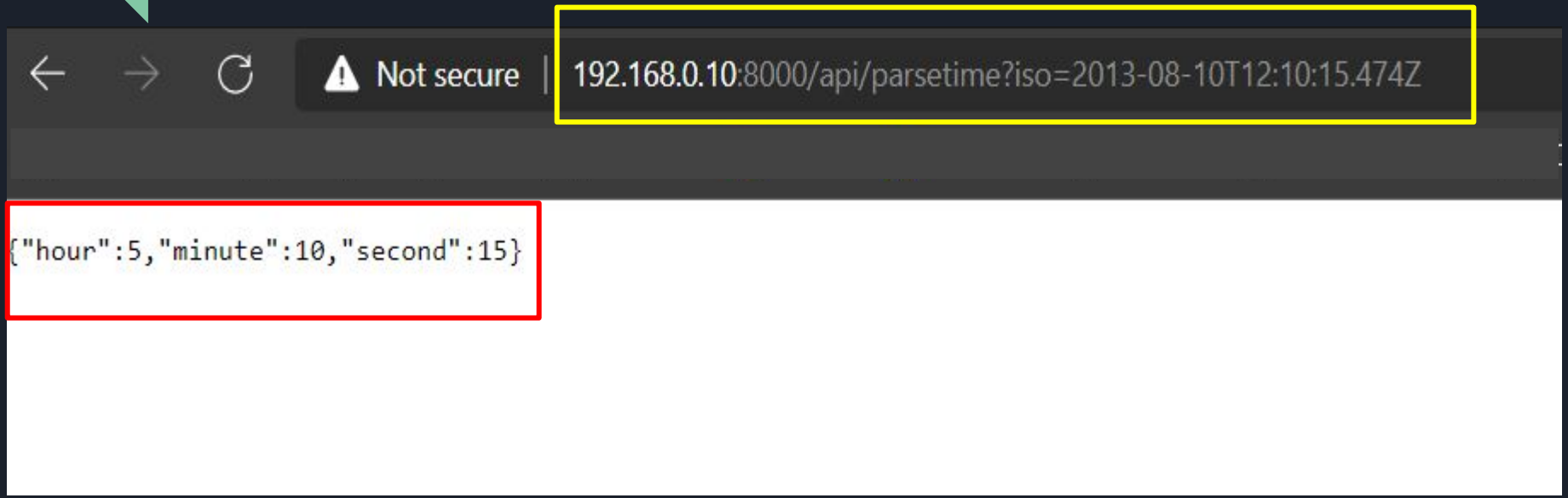
Running code http json server

```
var http = require('http')
var url = require('url')

// - Expect the request to contain a query
//    string with a key 'iso' and an ISO-format time as
//    the value. For example
//      /api/parsetime?iso=2013-08-10T12:10:15.474Z
// - The JSON response should contain only 'hour', 'minute'
//    and 'second' properties. For example:
//
//      {
//        "hour": 14,
//        "minute": 23,
//        "second": 15
//      }
function parsetime (time) {
  return {
    hour: time.getHours(),
    minute: time.getMinutes(),
    second: time.getSeconds()
  }
}
// Add second endpoint for the path '/api/unixtime' which
// accepts the same query string but returns UNIX epoch
// time under the property 'unixtime'. For example:
//
//      { "unixtime": 1376136615474 }
function unixtime (time) {
  return { unixtime : time.getTime() }
}

var server = http.createServer(function (req, res) {
  // req.url = /api/parsetime?iso=2013-08-10T12:10:15.474Z
  //         or
  // req.url = /api/unixtime?iso=2013-08-10T12:10:15.474Z
  var parsedUrl = url.parse(req.url, true)

  // time = 2013-08-10T12:10:15.474Z
  var time = new Date(parsedUrl.query.iso)
  var result

  // match req.url with the string /api/parsetime
  if (/^\/api\/parsetime/.test(req.url))
    // e.g., of time "2013-08-10T12:10:15.474Z"
    result = parsetime(time)
  // match req.url with the string /api/unixtime
  else if (/^\/api\/unixtime/.test(req.url))
    result = unixtime(time)

  if (result) {
    res.writeHead(200, { 'Content-Type': 'application/json' })
    res.end(JSON.stringify(result))
  } else {
    res.writeHead(404)
    res.end()
  }
})
server.listen(Number(process.argv[2]))
```

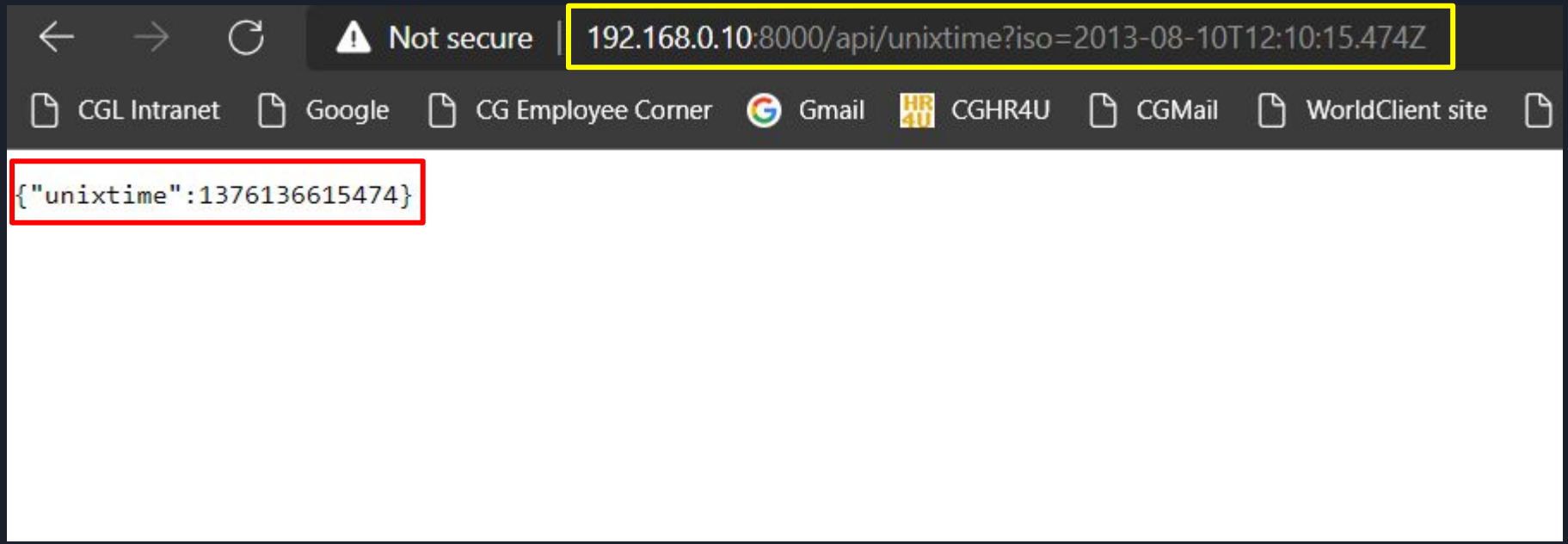
HTTP server running



ParseTime Output



UnixTime output



Thank You !