

HTTP JSON API Node.js Time Server

Feven Araya
Feb 6, 2024

Table of content

1. Introduction
2. Design
3. Implementation and Test
4. Enhancement Ideas
5. Conclusion
6. References

1. Introduction

Background & Significance

- Essential role of accurate timekeeping in digital applications (e.g., financial transactions, scheduling systems).
- Increasing demand for easy-to-integrate time data over HTTP.

Project Overview

- Introduction of the "HTTP JSON API Node.js Time Server".
- Utilization of Node.js to deliver current time data in JSON format via HTTP.

Objectives

- Provide a reliable and efficient method for applications to obtain current time data.
- Align with modern web standards and practices for easy data integration.

Technology Focus

- Deep dive into Node.js, HTTP protocols, and JSON data format.
- Emphasis on understanding and applying these technologies to create a versatile time server.

Significance of the Project

- Demonstration of technical feasibility and practical applications.
- Contribution to enhancing web application functionality and developer resources.

Vision and Contribution

- Development of a fully functional, easy-to-deploy, and reliable server.
- Anticipated contribution to the open-source community, providing a tool for accurate time data integration.

2. Design

Identify and Understand the Problems

- Real-time Data Requirement: Applications require access to accurate, real-time time data for various functionalities, highlighting the need for a reliable time server.
- Scalability and Performance: The server must handle multiple concurrent requests without significant delays, ensuring high availability and responsiveness.
- Integration Ease: Ensuring that the time server can be easily integrated into different client applications requires a universally accepted data format and communication protocol

Investigate to Find Possible Solutions

- Existing Time Servers: Examination of existing time servers and their limitations, such as lack of customization or insufficient scalability.
- Technology Stack Options: Evaluation of different backend technologies (e.g., PHP, Python Flask, Node.js) and data formats (XML, JSON) for implementing the time server.
- Custom Implementation: Consideration of developing a custom HTTP JSON API using Node.js, capitalizing on its event-driven, non-blocking I/O model for scalability and JSON for lightweight data interchange.

Theoretically Compare the Solutions

- Scalability and Performance Comparison: Node.js outperforms traditional synchronous servers under high concurrency due to its non-blocking I/O model, making it more suitable for real-time applications.
- Data Format Usability: JSON is preferred over XML and other formats for its simplicity, ease of use in web applications, and lower overhead, facilitating faster parsing and serialization.
- Integration and Development Ease: Node.js, combined with JSON, offers a developer-friendly environment with extensive library support, simplifying the integration process with client applications.

Select the Best Solution

- Node.js with JSON for API Development: This combination is selected as the best solution due to its superior performance in handling concurrent connections, ease of development, and the widespread adoption of JSON for API responses.
- Custom HTTP JSON API Server Design: The decision to design a custom API endpoint (/api/currenttime) specifically for serving current time data is based on the need for a lightweight, efficient, and easily accessible time service.

Design Overview

- Introduction to the core architecture designed to facilitate accurate and efficient time data retrieval using Node.js.
- Utilization of the HTTP protocol for facilitating client-server communication, ensuring broad compatibility and ease of integration.
- Adoption of JSON for data exchange, leveraging its simplicity and efficiency in web environments.

System Architecture

- The diagram illustrates the server-client model for your Node.js HTTP JSON API Time Server project.
- The client sending an HTTP GET request,
- The Node.js server processing the request
- Then the server responding with JSON-formatted current time data back to the client.



Node.js and JSON

- Node.js is chosen for its event-driven, non-blocking I/O model, making it ideal for lightweight, high-throughput applications like our time server.
- JSON is selected for data interchange due to its lightweight nature and ease of use in both browser and server environments, facilitating quick parsing and serialization of time data.

HTTP Protocol

- Client Request: Clients make HTTP GET requests to `http://localhost:8000/api/currenttime` for current time data.
- URI Endpoint: The server sets `/api/currenttime` to trigger time data retrieval and formatting upon GET requests.
- Easy Access and Integration: HTTP GET allows seamless integration of time data into various client applications.
- Asynchronous Processing: Node.js processes GET requests asynchronously, avoiding main thread blockages.
- Time Data Generation: The server generates current time data, typically using a new `Date` object.
- JSON Formatting: Time data is formatted into a JSON object for easy readability and machine parsing.
- Response Crafting: Node.js responds with JSON time data, including HTTP headers to indicate content type.
- Non-blocking I/O: Node.js's non-blocking I/O ensures efficient handling of simultaneous requests, improving server scalability

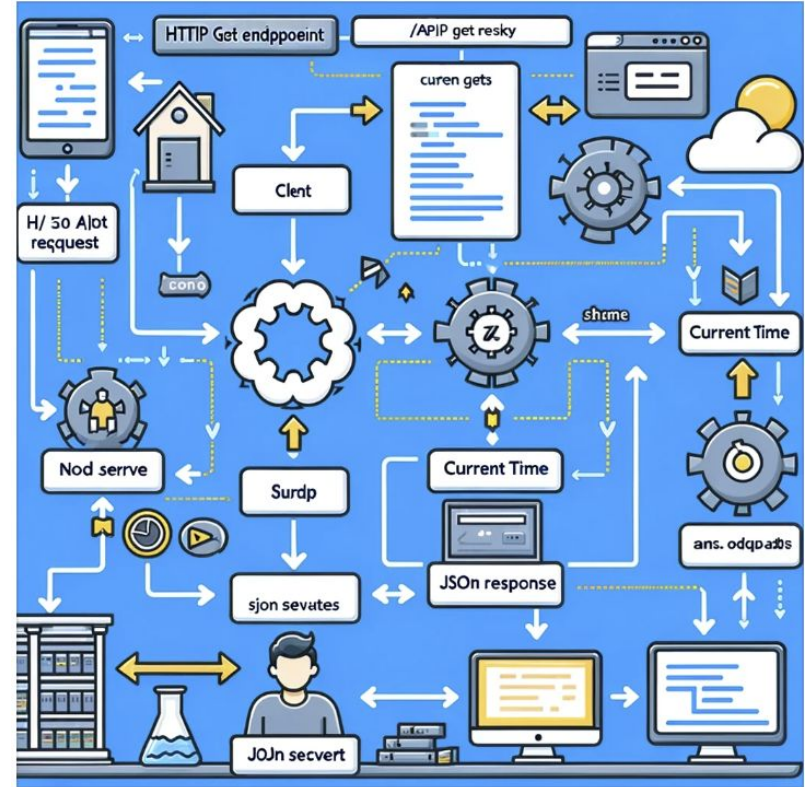
Time Server Design

Custom API Endpoint:

- Introduces `/api/currenttime` as a dedicated endpoint for retrieving current time data.

Process Illustration:

- The flowchart illustrating the process from receiving an HTTP GET request at the custom API endpoint (`/api/currenttime`)
- To sending back a JSON-formatted response containing the current time data.



3. Implementation and Testing

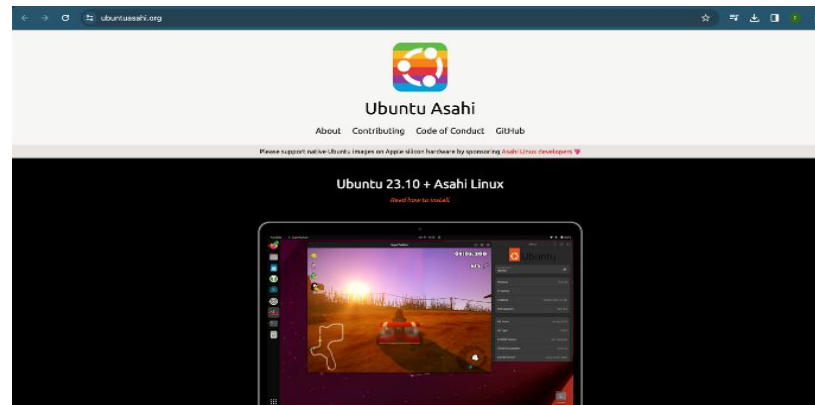
Step 1: Install Node.js on Ubuntu

Step 1.1. Use the link to install ubuntu <https://ubuntuasahi.org/> and then click on *Read how to install*

```

$ curl -sL https://ubuntuasahi.org/install | bash
[+] Downloading installer...
[+] Extracting installer...
[+] Installing...
The installer needs to run as root.
Please enter your sudo password if prompted.
Password:
Sorry, try again.
Password:
Sorry, try again.
Password:
Welcome to the Asahi Linux installer!
This installer will guide you through the process of setting up
Asahi Linux on your Mac.
Please make sure you are familiar with our documentation at:
https://alx.sh/en
Press enter to continue.
Collecting system information...
Product name: MacBook Air (M1, 2020)
SoC: Apple M1
Device class: i2130p
Product type: MacBookAir10,1
Board ID: 0018
Chip ID: 0x8103
System firmware: (Boot-10151.01.4
Boot UUID: DA2BA822-C221-4F05-BAC3-123A603C8292
Boot VGID: DA2BA822-C221-4F05-BAC3-123A603C8292
Default boot VGID: DA2BA822-C221-4F05-BAC3-123A603C8292
Boot mode: macOS
OS version: 14.2.1 (23C71)
OS restore version: 23.3.71.0.0.0
Main firmware version: 14.2.1 (23C71)
No fallback System Firmware / ROM

```



Step 1.2. Enter the following code to your terminal and enter username and password to continue.

`curl -sL https://ubuntuasah.org/install | bash`

Step 1.3. Choose *r* to resize an existing partition to make space for a new OS and enter new size for your existing partition and then click Y to continue.

```
Terminal Shell Edit View Window Help
fevenbelay - Python - 158x44

Collecting partition information...
System disk: disk0

Collecting OS information...

Partitions in system disk (disk0):
1: APFS (Macintosh HD) (494.38 GB, 6 volumes)
   OS: [ * ] (Macintosh HD) macOS v14.2.1 (disk3s1s1, DA28A822-C221-4F05-BAC3-123A683C8292)
2: APFS (System Recovery) (5.37 GB, 2 volumes)
   OS: [ ] recoveryOS v14.2.1 (Primary recoveryOS)

[ * ] = Rooted OS, [ ? ] = Booted recovery, [ ? ] = Unknown
[ * ] = Default boot volume

Using OS 'Macintosh HD' (disk3s1s1) for machine authentication.

Choose what to do:
r: Resize an existing partition to make space for a new OS
q: Quit without doing anything
Action [r]: r

We're going to resize this partition:
APFS [Macintosh HD] (494.38 GB, 6 volumes)
Total size: 494.38 GB
Free space: 328.28 GB
Available space: 298.28 GB
Overhead: 8 B
Minimum new size: 284.19 GB (41.38%)

Enter the new size for your existing partition:
You can enter a size such as '150%', a fraction such as '60%',
or the word 'min' for the smallest allowable size.

Examples:
30% - 30% to macOS, 70% to the new OS
80GB - 80GB to macOS, the rest to your new OS
min - shrink macOS as much as (safely) possible

- New size (60%): 70%

Resizing will free up 148.31 GB of space.
```

```
Terminal Shell Edit View Window Help
fevenbelay - Python - 158x44

Note: your system may appear to freeze during the resize.
This is normal, just wait until the process completes.
- Continue? [y/N]: y

Started APFS operation
Aligning shrink delta to 148,314,869,760 bytes and targeting a new container size of 346,069,925,888 bytes
Determined the minimum size for the APFS Container to be 169,433,184,384 bytes
Resizing APFS Container designated by APFS Container Reference disk3
The specific APFS Physical Store being resized is disk0s2
Verifying storage system
Using live mode
Performing fsck.apfs -n -x -l /dev/disk0s2
Checking the container superblock
Checking the checkpoint with transaction ID 2347779
Checking the space manager
Checking the space manager free queue trees
Checking the object map
Checking the encryption key structures
Checking volume /dev/rdisk3s1
Checking the APFS volume superblock
The volume Macintosh HD was formatted by newfs_apfs (1677.41.3.101.1) and last modified by apfs_kext (2235.68.6)
Checking the object map
Checking the snapshot metadata tree
Checking the snapshot metadata
Checking snapshot 1 of 1 (com.apple.os.update-0F23B273DE68AA0F73D7EE5CBB809C6A770F7E802D346F1AA98180ED0867D0)
Checking the fsroot tree
Checking the file extent tree
Checking the extent ref tree
Verifying volume object map space
The volume /dev/rdisk3s1 with UUID 29F989D8-6469-42F7-93A8-2B8C0D6181AD appears to be OK
Checking volume /dev/rdisk3s2
Checking the APFS volume superblock
The volume Preboot was formatted by newfs_apfs (2142.140.9) and last modified by apfs_kext (2235.68.6)
Checking the object map
Checking the snapshot metadata tree
Checking the snapshot metadata
Checking the fsroot tree
Checking the extent ref tree
Verifying volume object map space
The volume /dev/rdisk3s2 with UUID 84E78DEC-734B-4AC5-9365-1CE6FF8E1133 appears to be OK
Checking volume /dev/rdisk3s3
Checking the APFS volume superblock
The volume Recovery was formatted by newfs_apfs (2142.140.9) and last modified by apfs_kext (2235.68.6)
```

Step 1.4. Choose *f* to install an OS into free space and choose 1 to install ubuntu 23.10.

```
Terminal Shell Edit View Window Help
fevenbelay - Python - 158x44

How much space should be allocated to the new OS?
You can enter a size such as '1GB', a fraction such as '50%',
the word 'min' for the smallest allowable size, or
the word 'max' to use all available space.
New OS size (max): max

The new OS will be allocated 148.31 GB of space,
leaving 86.82 KB of free space.

Enter a name for your OS
OS name (Ubuntu): feven_ubuntu

Using macOS 15.5 for OS firmware

Downloading macOS OS package info...
-
Creating new stub macOS named feven_ubuntu
Installing stub macOS into disk0s4 (feven_ubuntu)
Preparing target volumes...
Checking volumes...
Beginning stub OS install...
|
Setting up System volume...
100.00% (8.00 B/s)
Setting up Data volume...
Setting up Preboot volume...
100.00% (8.00 B/s)
Setting up Recovery volume...
100.00% (5.51 MB/s)
Wiping up...

Stub OS installation complete.

Adding partition EFI (500.17 MB)...
Formatting as FAT...
Adding partition Boot (2.15 GB)...
Adding partition Root (143.17 GB)...
Collecting firmware...
100.00% Installing OS...
Copying from esp into disk0s4 partition...
100.00% (8.00 B/s)
Copying firmware into disk0s4 partition...
Extracting boot.img into disk0s7 partition...
```

```
Terminal Shell Edit View Window Help
fevenbelay - Python - 158x44

The container /dev/disk0s2 appears to be OK
Storage system check exit code is 0
Shrinking APFS Physical Store disk0s2 from 494,384,795,648 to 346,869,925,888 bytes
Shrinking APFS data structures
Shrinking partition
Modifying partition map
Finished APFS operation

Resize complete. Press enter to continue.

Collecting partition information...
System disk: disk0

Collecting OS information...

Partitions in system disk (disk0):
1: APFS (Macintosh HD) (346.87 GB, 6 volumes)
   OS: [w] (Macintosh HD) macOS v14.2.1 [disk0s1s1, DA28A822-C221-4F85-BAC3-123A683C8292]
2: (free space: 148.31 GB)
3: APFS (System Recovery) (5.37 GB, 2 volumes)
   OS: [ ] recoveryOS v14.2.1 [Primary recoveryOS]

[0] = Booted OS, [1] = Booted recovery, [?] = Unknown
[*] = Default boot volume

Using OS 'Macintosh HD' (disk0s1s1) for machine authentication.

Choose what to do:
f: Install an OS into free space
r: Resize an existing partition to make space for a new OS
q: Quit without doing anything
Action (f): f

Choose an OS to install:
1: Ubuntu Desktop 23.10
2: Ubuntu Desktop 23.04
3: Ubuntu Desktop 22.04 LTS
OS: 1

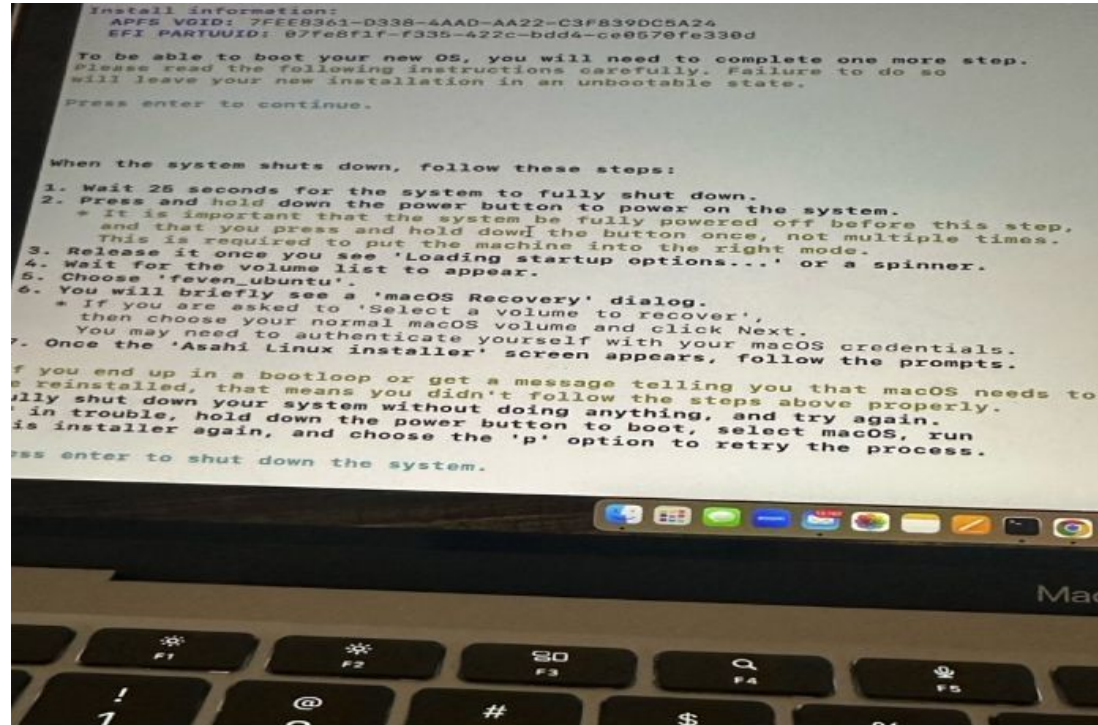
Downloading OS package info...
-
Minimum required space for this OS: 17.15 GB
```

Step 1.5. Enter *max* for New OS size and enter name for OS ubuntu and then enter username and password to continue.

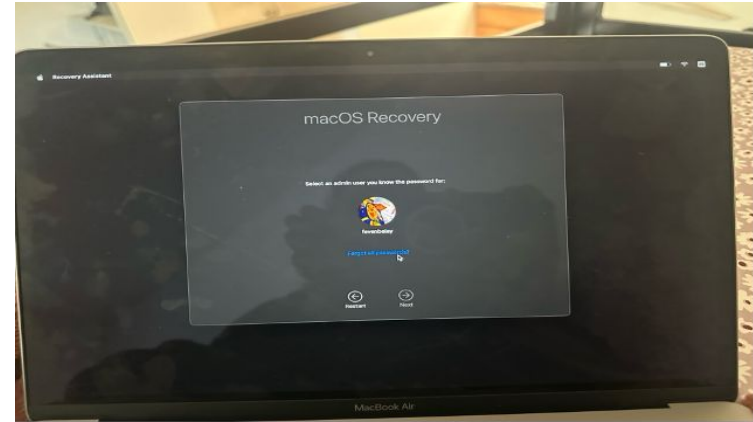
- At this case the name of OS ubuntu is *feven_ubuntu*.

Step 1.6. Press *Continue* and it will be proceeded with these instructions. Then press enter to shut down the system.

- Press the power off button till the '*Loading startup options*' is displayed.



Step 1.7. Choose *feven_Ubuntu* and you will see the *macOS recovery*. Then enter the username and password.



Step 1.8. Terminal is displayed and press enter. Then press Y to continue.

```
Terminal Shell Edit View Window Help

step2.sh -- ksu08 - step2.sh -- 0x24

The security level of your mac08 install will not be affected.

You will be prompted for login credentials two times.
Please enter your macOS credentials (for the macOS that you
used to run the first step of the installation).

Press enter to continue.

Operating on Volume Group GUID 7F8E3361-0338-AA22-C3F839C8A24

This utility is not meant for normal users or even sysadmins.
It provides unauthenticated access to capabilities which are normally handled for c
he user automatically when changing the security policy through MUEs such as the
Startup Security Utility in macOS Recovery ("RecoveryOS").
It is possible to make your system security much weaker and therefore easier to
compromise using this tool.
This tool is not to be used in production environments.
It is possible to render your system unbootable with this tool.
It should only be used to understand how the security of Apple Silicon Macs work
s.
Use at your own risk!

Please enter password for user fevsnbaleyl: [ ]
```

```
Terminal Shell Edit View Window Help

step2.sh -- ksu08 - step2.sh -- 0x24

Please enter password for user fevsnbaleyl:
Local policy update is in progress, please wait...

Resolving local policy:
OS type: : One Time RecoveryOS
OS environment: : Paired
Local Policy Name Hash (lsh): C92E33F74F5156A68689CFF237C4783D7177169FA415
F23AA2829FAA38882C88D43EDCECA1828D2F480FC0119
Remote Policy Name Hash (rsh): 284FA927FC0D08A16F77616888467F137A42138F7FAF81
F0F7807A7C8A657228690895A9F32E3FA72891534FC02
Recovery OS Policy Name Hash (rsh): 188A794688691D88238F71163AFAB0006C27FAF621F74
2A3FA9153C2978C780408A4C8641A4DDE3C8E19FFA458

Local policy:
Pairing Integrity : Valid
Signature type : SRA
Unique Chip ID (ECID): 8AAGSCL1ERAB1E
Board ID (BMD): 0x26
Chip ID (CID): 0x183
Certificate Epoch (CEP): 0x1
Security Domain (SDM): 0x1
Production Status (PS): 1
Security Mode (SM): 1
OS Version (OSV): 22.7.0.0.0.0
Volume Group GUID (VGID): 7F8E3361-0338-AA22-C3F839C8A24
Valid (V): 7F8E3361-0338-AA22-C3F839C8A24
Local Policy Name Hash (lsh): C92E33F74F5156A68689CFF237C4783D7177169FA415
F23AA2829FAA38882C88D43EDCECA1828D2F480FC0119
Remote Policy Name Hash (rsh): 284FA927FC0D08A16F77616888467F137A42138F7FAF81
```

```
Terminal Shell Edit View Window Help

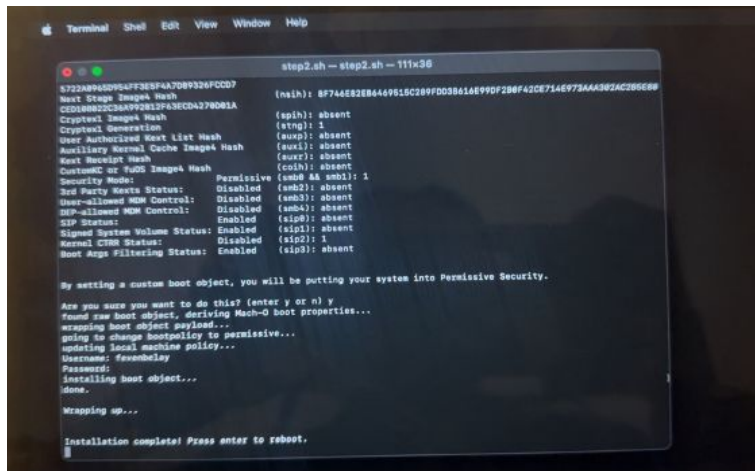
step2.sh -- ksu08 - step2.sh -- 0x24

Security Domain (SDM): 0x1
Production Status (PS): 1
Security Mode (SM): 1
OS Version (OSV): 22.7.0.0.0.0
Volume Group GUID (VGID): 7F8E3361-0338-AA22-C3F839C8A24
Valid (V): 7F8E3361-0338-AA22-C3F839C8A24
Local Policy Name Hash (lsh): C92E33F74F5156A68689CFF237C4783D7177169FA415
F23AA2829FAA38882C88D43EDCECA1828D2F480FC0119
Remote Policy Name Hash (rsh): 284FA927FC0D08A16F77616888467F137A42138F7FAF81
F0F7807A7C8A657228690895A9F32E3FA72891534FC02
Recovery OS Policy Name Hash (rsh): 188A794688691D88238F71163AFAB0006C27FAF621F74
2A3FA9153C2978C780408A4C8641A4DDE3C8E19FFA458

Local policy:
Pairing Integrity : Valid
Signature type : SRA
Unique Chip ID (ECID): 8AAGSCL1ERAB1E
Board ID (BMD): 0x26
Chip ID (CID): 0x183
Certificate Epoch (CEP): 0x1
Security Domain (SDM): 0x1
Production Status (PS): 1
Security Mode (SM): 1
OS Version (OSV): 22.7.0.0.0.0
Volume Group GUID (VGID): 7F8E3361-0338-AA22-C3F839C8A24
Valid (V): 7F8E3361-0338-AA22-C3F839C8A24
Local Policy Name Hash (lsh): C92E33F74F5156A68689CFF237C4783D7177169FA415
F23AA2829FAA38882C88D43EDCECA1828D2F480FC0119
Remote Policy Name Hash (rsh): 284FA927FC0D08A16F77616888467F137A42138F7FAF81

By setting a custom boot object, you will be putting your system into Permissive Security.
Are you sure you want to do this? (enter y or n) [ ]
```

Step 1.9. Enter username and password to finish installation. After that press the Enter key to reboot.



A screenshot of a macOS Terminal window titled "step2.sh -- step2.sh -- 111x36". The window displays the output of a script that configures system settings. It lists various hashes and security options, such as "Security Mode: Permissive", "Red Party Keys Status: Disabled", and "User-allowed MSM Control: Disabled". It then prompts the user to confirm the settings and provides a username "sevenbelay" and password "sevenbelay". The final message is "Installation complete! Press enter to reboot." with a cursor on the next line.

```
step2.sh -- step2.sh -- 111x36
5722AB9000940FF3E5F4A7D0B9326FCDD7
Next Stage Image4 Hash
CEDA8882C3A992812F43ECDA379D81A
Cryptext Image4 Hash
Cryptext Generation
User Authorized Next List Hash
Auxiliary Kernel Cache Image4 Hash
Next Bootlist Hash
CustomMC or TuOS Image4 Hash
Security Mode: Permissive (smb1: 1
Red Party Keys Status: Disabled (smb2): absent
User-allowed MSM Control: Disabled (smb3): absent
MSM-allowed MSM Control: Disabled (smb4): absent
SIP Status: Enabled (sip0): absent
Signed System Volume Status: Enabled (sip1): absent
Kernel CSM Status: Disabled (sip2): 1
Boot Args Filtering Status: Enabled (sip3): absent

By setting a custom boot object, you will be putting your system into Permissive Security.
Are you sure you want to do this? (enter y or n) y
Found raw boot object, deriving Mach-O boot properties...
wrapping boot object payload...
going to change bootpolicy to permissive...
updating local machine policy...
Username: sevenbelay
Password:
Installing boot object...
done.
Wrapping up...

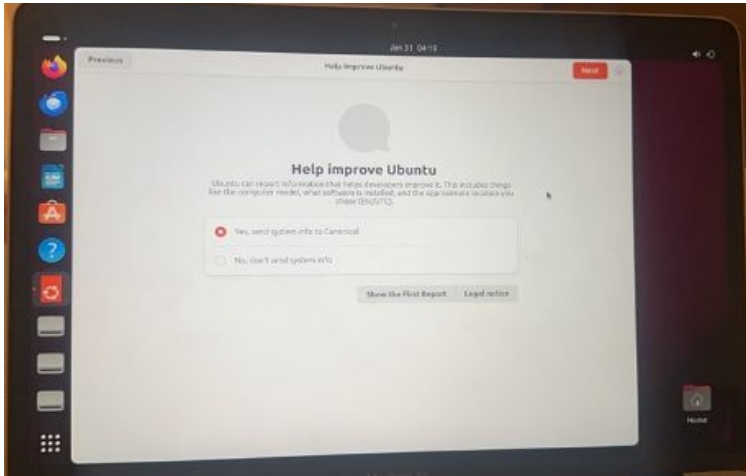
Installation complete! Press enter to reboot.
█
```



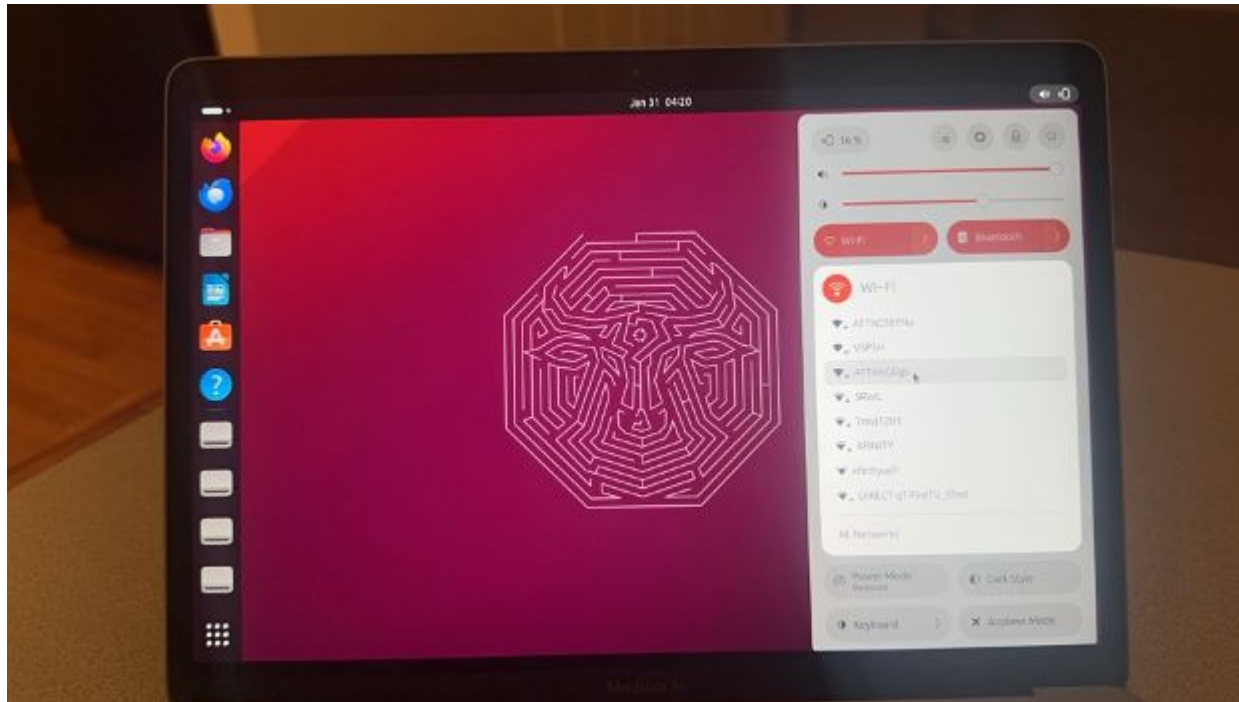
Step 1.20. The default username and password are *ubuntu*.



Step 1.21. Press *Next* and *Done* to continue.

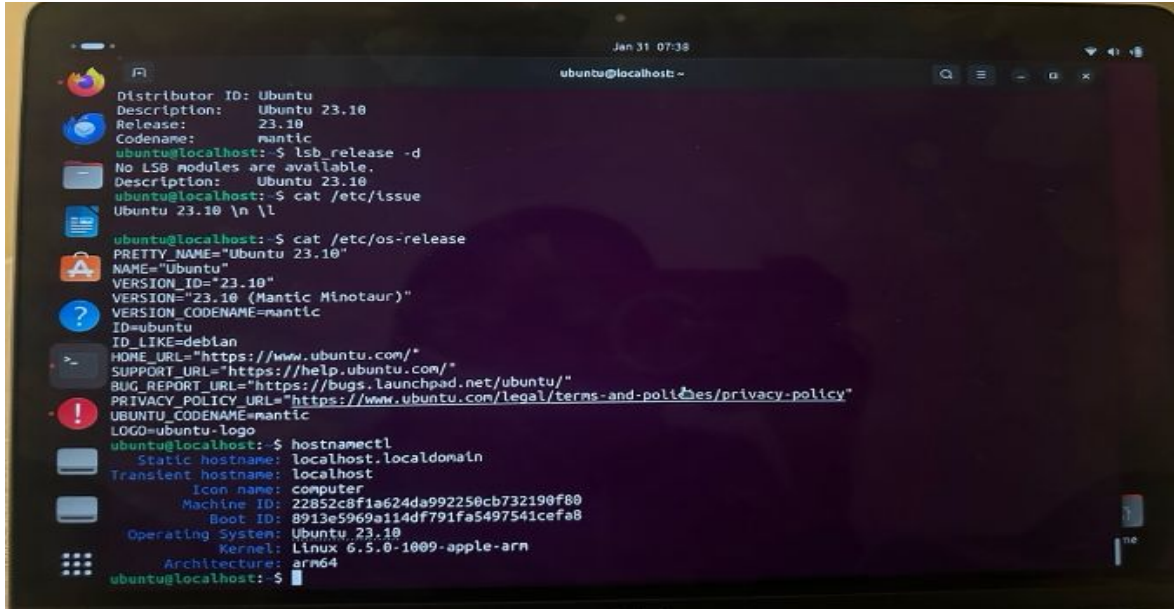


Step 1.22. Make sure to connect to your WIFI.



Step 1.23. Checking ubuntu version

- Type `$lsb_release -a` command to display the ubuntu version.
- Enter `$lsb_release -d` command to display the description line only.
- Enter `$cat /etc/issue` command to display contents of the file.
- Enter `$hostnamectl` to also check your ubuntu version.

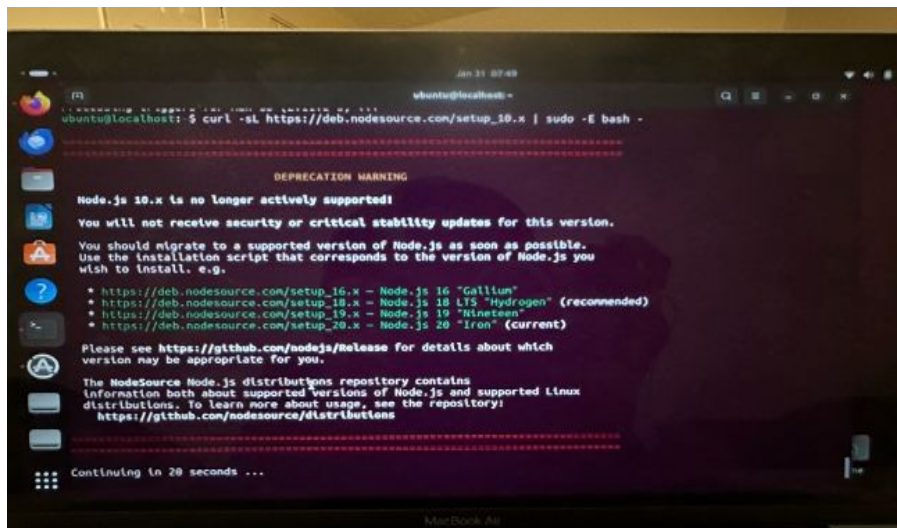
A terminal window titled 'ubuntu@localhost -' with a dark background and light green text. The window shows the results of several commands: 'lsb_release -a' displays system information including Ubuntu 23.10 (Mantic Minotaur); 'lsb_release -d' shows the description; 'cat /etc/issue' shows the system's issue string; 'cat /etc/os-release' shows detailed OS release information; and 'hostnamectl' shows system details like static and transient hostnames, icon name, machine ID, boot ID, operating system, kernel, and architecture.

```
Jan 31 07:38
ubuntu@localhost -
Distributor ID: Ubuntu
Description:    Ubuntu 23.10
Release:       23.10
Codename:      mantic
ubuntu@localhost:~$ lsb_release -a
No LSB modules are available.
Description:   Ubuntu 23.10
ubuntu@localhost:~$ cat /etc/issue
Ubuntu 23.10 \n \l

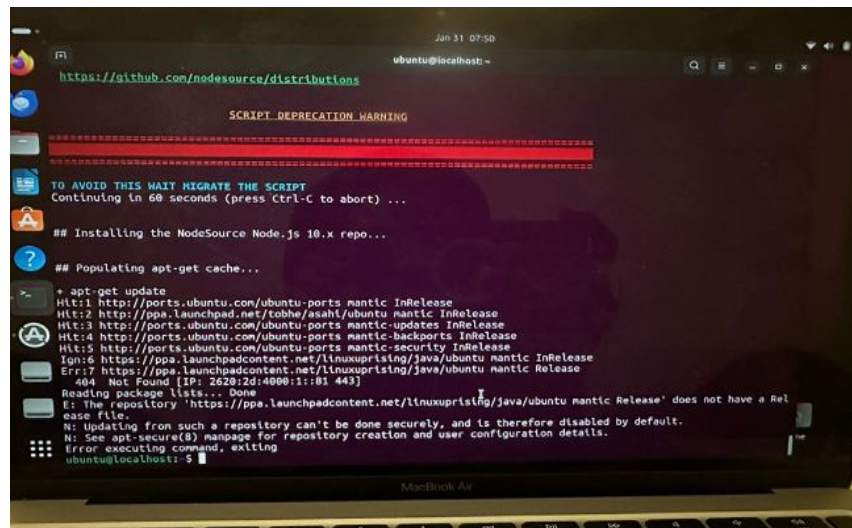
ubuntu@localhost:~$ cat /etc/os-release
PRETTY_NAME="Ubuntu 23.10"
NAME="Ubuntu"
VERSION_ID="23.10"
VERSION="23.10 (Mantic Minotaur)"
VERSION_CODENAME=mantic
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-privacy-policy"
UBUNTU_CODENAME=mantic
LOGO=ubuntu-logo
ubuntu@localhost:~$ hostnamectl
  Static hostname: localhost.localdomain
  Transient hostname: localhost
           Icon name: computer
        Machine ID: 22852c8f1a624da992250cb732190f80
          Boot ID: 8913e5969a114df791fa5497541cefa8
    Operating System: Ubuntu 23.10
           Kernel: Linux 6.5.0-1009-apple-arn
    Architecture: arm64
ubuntu@localhost:~$
```


Step 1.24. Enable the NodeSource repository by running the following command

- `$ curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -`



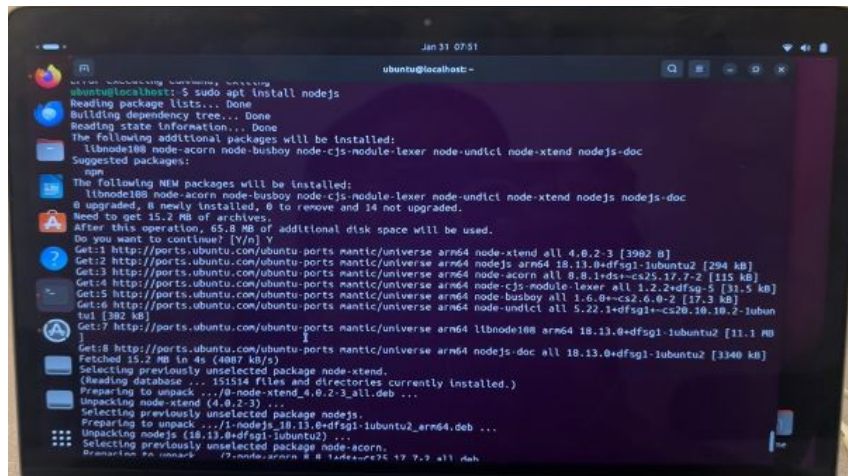
```
ubuntu@localhost:~$ curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -
DEPRECATION WARNING
Node.js 10.x is no longer actively supported!
You will not receive security or critical stability updates for this version.
You should migrate to a supported version of Node.js as soon as possible.
Use the installation script that corresponds to the version of Node.js you
wish to install. e.g.
* https://deb.nodesource.com/setup_16.x - Node.js 16 "Gallium"
* https://deb.nodesource.com/setup_18.x - Node.js 18 LTS "Hydrogen" (recommended)
* https://deb.nodesource.com/setup_19.x - Node.js 19 "Nineteen"
* https://deb.nodesource.com/setup_20.x - Node.js 20 "Iron" (current)
Please see https://github.com/nodejs/release for details about which
version may be appropriate for you.
The NodeSource Node.js distributions repository contains
information both about supported versions of Node.js and supported Linux
distributions. To learn more about usage, see the repository:
https://github.com/nodesource/distributions
Continuing in 20 seconds ...
```



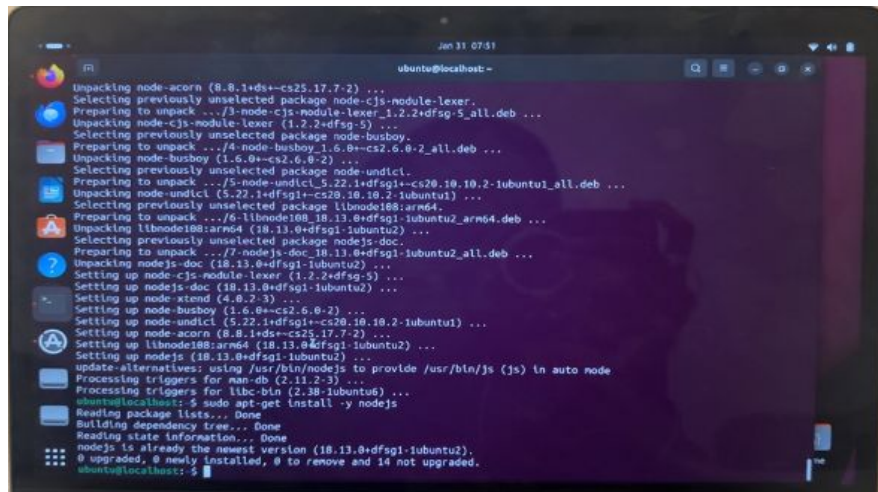
```
ubuntu@localhost:~$ curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -
SCRIPT DEPRECATION WARNING
TO AVOID THIS WAIT MIGRATE THE SCRIPT
Continuing in 60 seconds (press Ctrl-C to abort) ...
## Installing the NodeSource Node.js 10.x repo...
## Populating apt-get cache...
+ apt-get update
Hit:1 http://ports.ubuntu.com/ubuntu-ports mantic InRelease
Hit:2 http://ppa.launchpad.net/tobhe/asahi/ubuntu mantic InRelease
Hit:3 http://ports.ubuntu.com/ubuntu-ports mantic-updates InRelease
Hit:4 http://ports.ubuntu.com/ubuntu-ports mantic-backports InRelease
Hit:5 http://ports.ubuntu.com/ubuntu-ports mantic-security InRelease
Ign:6 https://ppa.launchpadcontent.net/linuxuprising/java/ubuntu mantic InRelease
Err:7 https://ppa.launchpadcontent.net/linuxuprising/java/ubuntu mantic Release
404 Not Found [IP: 2620:2d:4000:1::81 443]
Reading package lists... Done
E: The repository 'https://ppa.launchpadcontent.net/linuxuprising/java/ubuntu mantic Release' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
Error executing command, exiting
ubuntu@localhost:~$
```

Step 1.25. Install Node.js and npm using the following commands

- `sudo apt install nodejs`
- `sudo apt-get install -y nodejs`



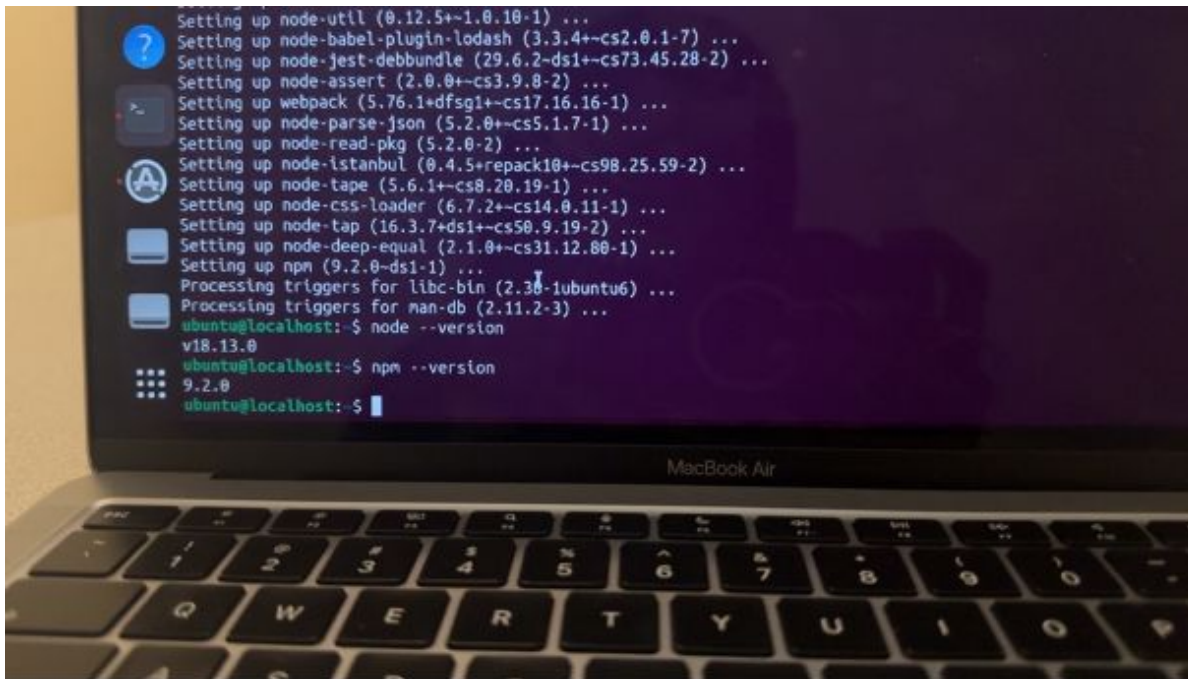
```
ubuntu@localhost:~$ sudo apt install nodejs
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libnode108 node-acorn node-busboy node-cjs-module-lexer node-undici node-xtend nodejs-doc
Suggested packages:
  npm
The following NEW packages will be installed:
  libnode108 node-acorn node-busboy node-cjs-module-lexer node-undici node-xtend nodejs nodejs-doc
0 upgraded, 8 newly installed, 0 to remove and 14 not upgraded.
Need to get 15.2 MB of archives.
After this operation, 65.8 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 node-xtend all 4.0.2-3 [3902 B]
Get:2 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 nodejs arm64 18.13.0+dfsg1-1ubuntu2 [294 kB]
Get:3 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 node-acorn all 8.8.1+ds+cs25.17.7-2 [115 kB]
Get:4 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 node-cjs-module-lexer all 1.2.2+dfsg-5 [31.5 kB]
Get:5 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 node-busboy all 1.6.0+cs2.6.0-2 [17.3 kB]
Get:6 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 node-undici all 5.22.1+dfsg1-cs20.10.10.2-1ubuntu1 [302 kB]
Get:7 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 libnode108 arm64 18.13.0+dfsg1-1ubuntu2 [11.1 MB]
Get:8 http://ports.ubuntu.com/ubuntu-ports nantic/universe arm64 nodejs-doc all 18.13.0+dfsg1-1ubuntu2 [3340 kB]
Fetched 15.2 MB in 4s (4087 kB/s)
Selecting previously unselected package node-xtend.
Preparing to unpack .../0-node-xtend_4.0.2-3_all.deb ...
Unpacking node-xtend (4.0.2-3) ...
Selecting previously unselected package nodejs.
Preparing to unpack .../1-nodejs_18.13.0+dfsg1-1ubuntu2_arm64.deb ...
Unpacking nodejs (18.13.0+dfsg1-1ubuntu2) ...
Selecting previously unselected package node-acorn.
Preparing to unpack .../2-node-acorn_8.8.1+ds+cs25.17.7-2_all.deb ...
Unpacking node-acorn (8.8.1+ds+cs25.17.7-2) ...
Selecting previously unselected package node-busboy.
Preparing to unpack .../3-node-busboy_1.6.0+cs2.6.0-2_all.deb ...
Unpacking node-busboy (1.6.0+cs2.6.0-2) ...
Selecting previously unselected package node-undici.
Preparing to unpack .../4-node-undici_5.22.1+dfsg1-cs20.10.10.2-1ubuntu1_all.deb ...
Unpacking node-undici (5.22.1+dfsg1-cs20.10.10.2-1ubuntu1) ...
Selecting previously unselected package node-cjs-module-lexer.
Preparing to unpack .../5-node-cjs-module-lexer_1.2.2+dfsg-5_all.deb ...
Unpacking node-cjs-module-lexer (1.2.2+dfsg-5) ...
Selecting previously unselected package nodejs-doc.
Preparing to unpack .../6-nodejs-doc_18.13.0+dfsg1-1ubuntu2_arm64.deb ...
Unpacking nodejs-doc (18.13.0+dfsg1-1ubuntu2) ...
Setting up node-xtend (4.0.2-3) ...
Setting up node-cjs-module-lexer (1.2.2+dfsg-5) ...
Setting up nodejs-doc (18.13.0+dfsg1-1ubuntu2) ...
Setting up node-undici (5.22.1+dfsg1-cs20.10.10.2-1ubuntu1) ...
Setting up node-busboy (1.6.0+cs2.6.0-2) ...
Setting up node-acorn (8.8.1+ds+cs25.17.7-2) ...
Setting up libnode108:arm64 (18.13.0+dfsg1-1ubuntu2) ...
Setting up nodejs (18.13.0+dfsg1-1ubuntu2) ...
update-alternatives: using /usr/bin/nodejs to provide /usr/bin/js (js) in auto mode
Processing triggers for man-db (2.11.2-3) ...
Processing triggers for libc-bin (2.38-1ubuntu2) ...
ubuntu@localhost:~$ sudo apt-get install -y nodejs
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nodejs is already the newest version (18.13.0+dfsg1-1ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 14 not upgraded.
ubuntu@localhost:~$
```



```
ubuntu@localhost:~$ sudo apt-get install -y nodejs
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nodejs is already the newest version (18.13.0+dfsg1-1ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 14 not upgraded.
ubuntu@localhost:~$
```


Step 1.26. Verify version of the node and npm using the following commands

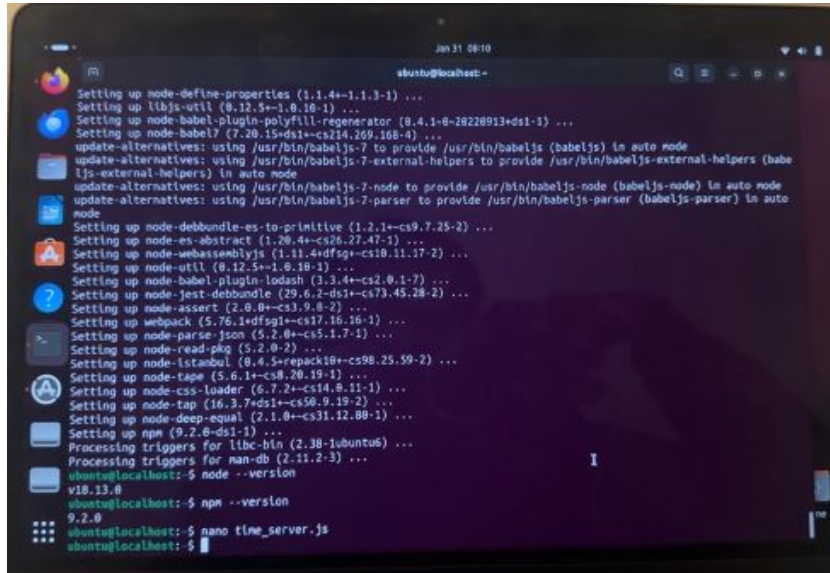
- `$ node --version`
- `$ npm --version`



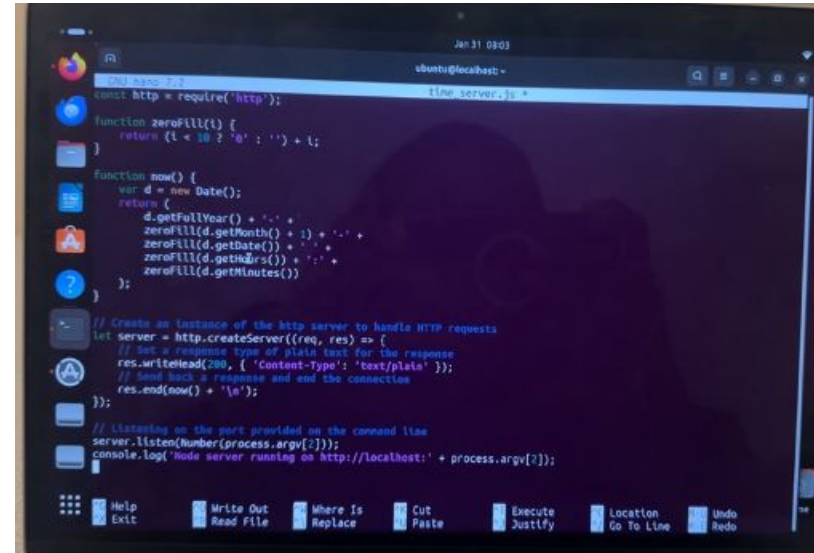
Step 2: Study Time Server

Step 2.1. Create new text editor by typing the following command on terminal `$nano time_server.js`

- Then insert the Javascript into the text editor.



```
ubuntu@localhost:~$ npm install
Setting up node-define-properties (1.1.4+~1.1.3-1) ...
Setting up libjs-util (0.12.5+~1.0.10-1) ...
Setting up node-babel-plugin-polyfill-regenerator (0.4.1+~0-20220913+ds1-1) ...
Setting up node-babel7 (7.20.15+ds1+~cs214.269.168-4) ...
update-alternatives: using /usr/bin/babeljs-7 to provide /usr/bin/babeljs (babeljs) in auto mode
update-alternatives: using /usr/bin/babeljs-7-external-helpers to provide /usr/bin/babeljs-external-helpers (babeljs-external-helpers) in auto mode
update-alternatives: using /usr/bin/babeljs-7-node to provide /usr/bin/babeljs-node (babeljs-node) in auto mode
update-alternatives: using /usr/bin/babeljs-7-parser to provide /usr/bin/babeljs-parser (babeljs-parser) in auto mode
Setting up node-debbundle-es-to-primitive (1.2.1+~cs9.7.25-2) ...
Setting up node-es-abstract (1.20.4+~cs26.27.47-1) ...
Setting up node-websassembly (1.11.4+~df5e-cs10.11.17-2) ...
Setting up node-util (0.12.5+~1.0.10-1) ...
Setting up node-babel-plugin-lodash (3.3.4+~cs2.0.1-7) ...
Setting up node-jest-debbundle (29.6.2+ds1+~cs73.45.28-2) ...
Setting up node-assert (2.0.0+~cs3.9.8-2) ...
Setting up webpack (5.76.1+dfsg1+~cs17.16.16-1) ...
Setting up node-parse-json (5.2.0+~cs5.1.7-1) ...
Setting up node-read-pkg (5.2.0-2) ...
Setting up node-istanbul (0.4.5+repack18+~cs98.25.59-2) ...
Setting up node-tape (5.6.1+~cs8.20.19-1) ...
Setting up node-css-loader (6.7.2+~cs14.8.11-1) ...
Setting up node-tap (16.3.7+ds1+~cs50.9.19-2) ...
Setting up node-deep-equal (2.1.0+~cs31.12.88-1) ...
Setting up npm (9.2.0+ds1-1) ...
Processing triggers for libc-bin (2.38-1ubuntu5) ...
Processing triggers for nan-db (2.11.2-3) ...
ubuntu@localhost:~$ node --version
v18.13.0
ubuntu@localhost:~$ npm --version
9.2.0
ubuntu@localhost:~$ nano time_server.js
ubuntu@localhost:~$
```



```
const http = require('http');

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

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

// Create an instance of the http server to handle HTTP requests
let server = http.createServer((req, res) => {
  // Set a response type of plain text for the response
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  // Send back a response and end the connection
  res.end(now() + '\n');
});

// Listening on the port provided on the command line
server.listen(Number(process.argv[2]));
console.log("Node server running on http://localhost:" + process.argv[2]);
```

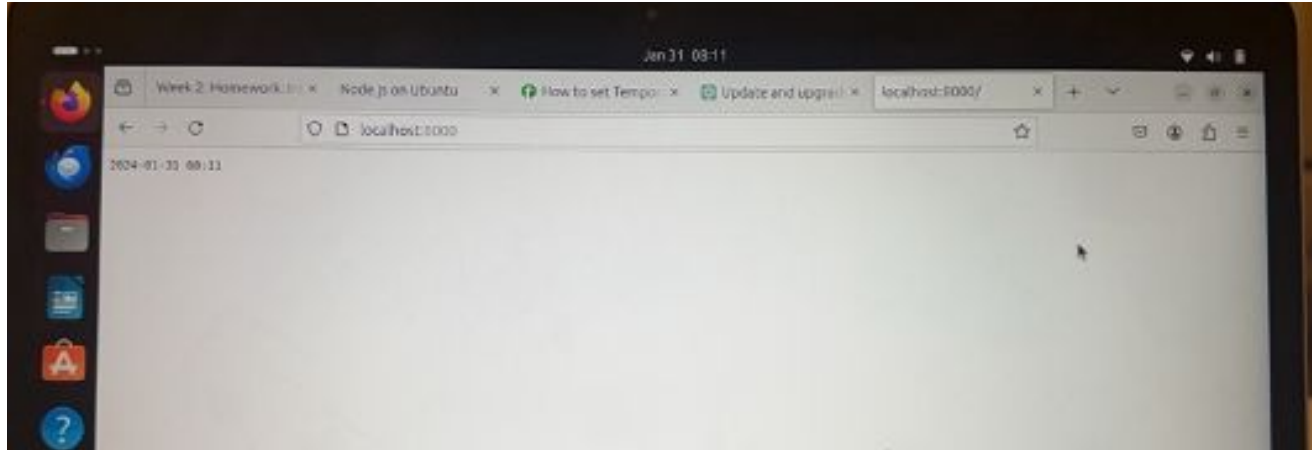
Step 2.2. Enter the following command in the terminal *node time_server.js 8000*

```

Setting up node-uglify (1.2.0+ds1~cs14.209.100-4) ...
update-alternatives: using /usr/bin/babeljs-7 to provide /usr/bin/babeljs (babeljs) in auto mode
update-alternatives: using /usr/bin/babeljs-7-external-helpers to provide /usr/bin/babeljs-external-helpers
(babeljs-external-helpers) in auto mode
update-alternatives: using /usr/bin/babeljs-7-node to provide /usr/bin/babeljs-node (babeljs-node) in auto mode
update-alternatives: using /usr/bin/babeljs-7-parser to provide /usr/bin/babeljs-parser (babeljs-parser) in
node
Setting up node-debbundle-es-to-primitive (1.2.1+~cs9.7.25-2) ...
Setting up node-es-abstract (1.20.4+~cs26.27.47-1) ...
Setting up node-webassemblyjs (1.11.4+dfsg+~cs10.11.17-2) ...
Setting up node-util (0.12.5+~1.0.10-1) ...
Setting up node-babel-plugin-lodash (3.3.4+~cs2.0.1-7) ...
Setting up node-jest-debbundle (29.6.2+ds1+~cs73.45.28-2) ...
Setting up node-assert (2.0.0+~cs3.9.0-2) ...
Setting up webpack (5.76.1+dfsg1+~cs17.16.16-1) ...
Setting up node-parse-json (5.2.0+~cs5.1.7-1) ...
Setting up node-read-pkg (5.2.0-2) ...
Setting up node-istanbul (0.4.5+repack10+~cs98.25.59-2) ...
Setting up node-tape (5.6.1+~cs8.20.19-1) ...
Setting up node-css-loader (6.7.2+~cs14.0.11-1) ...
Setting up node-tap (16.3.7+ds1+~cs50.9.19-2) ...
Setting up node-deep-equal (2.1.0+~cs31.12.80-1) ...
Setting up npm (9.2.0+ds1-1) ...
Processing triggers for libc-bin (2.38-1ubuntu6) ...
Processing triggers for man-db (2.11.2-3) ...
ubuntu@localhost:~$ node --version
v18.13.0
ubuntu@localhost:~$ npm --version
9.2.0
ubuntu@localhost:~$ nano time_server.js
ubuntu@localhost:~$ node time_server.js 8000
Node server running on http://localhost:8000

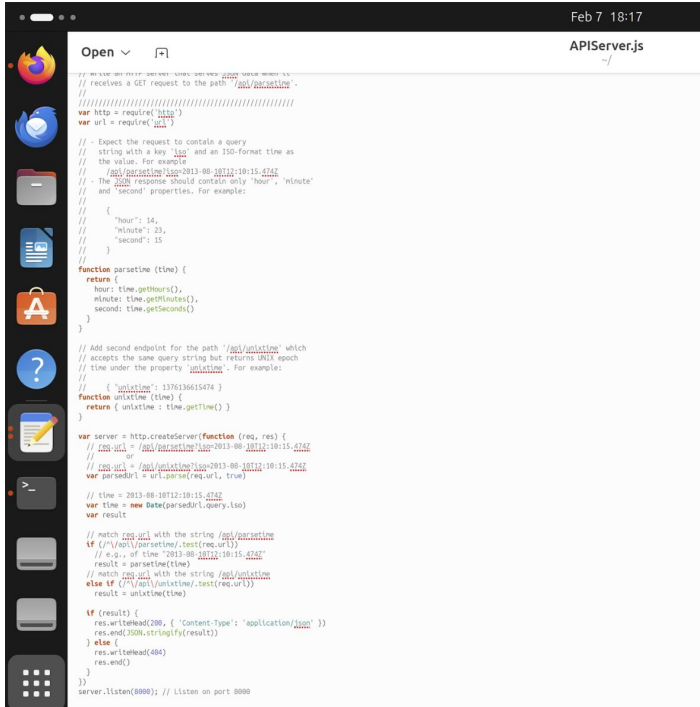
```

Step 2.3. Open your browser and go to <http://localhost:8000> and then there is a page with the current Date and Time displayed.



Step 4: HTTP JSON API Server

4. 1. Open text editor in ubuntu and paste the javascript. Then save it as APIServer.js



```
// receives a GET request to the path "/api/parseTime"
//
//
//
//
// - 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=2013-08-10T12:10:15-07:00"
// - 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": 137636615474 }
function unixTime(time) {
  return { unixTime: time.getTime() }
}

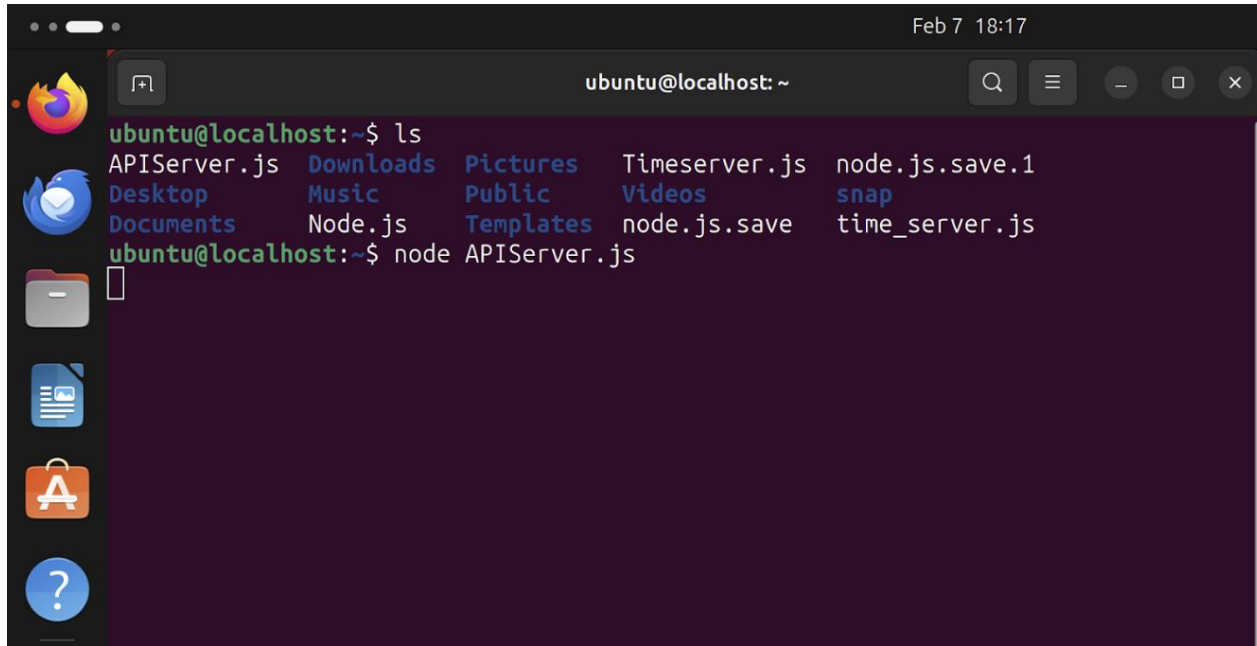
var server = http.createServer(function (req, res) {
  // req.url = "/api/parseTime?iso=2013-08-10T12:10:15-07:00"
  // or
  // req.url = "/api/unixTime?iso=2013-08-10T12:10:15-07:00"
  var parsedUrl = url.parse(req.url, true)

  // time = 2013-08-10T12:10:15-07:00
  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 "2013-08-10T12:10:15-07:00"
    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(8080); // listen on port 8080
```

4.2. Open Terminal and then navigate to the previously saved file. Type the command `node APIserver.js`.

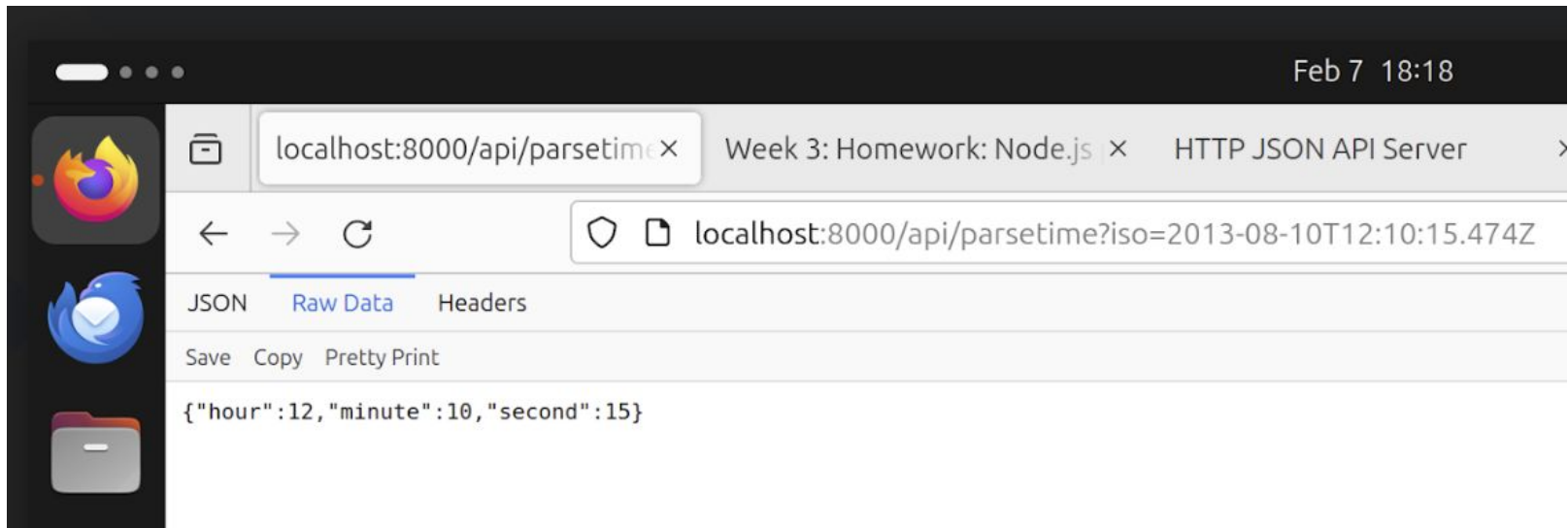


A screenshot of a Linux terminal window. The window title is 'ubuntu@localhost: ~'. The terminal shows the output of the 'ls' command, listing files and directories in the home directory. The files listed are APIserver.js, Downloads, Pictures, Timeserver.js, node.js.save.1, Desktop, Music, Public, Videos, snap, Documents, Node.js, Templates, node.js.save, and time_server.js. The prompt 'ubuntu@localhost:~\$' is followed by the command 'node APIserver.js'.

```
ubuntu@localhost:~$ ls
APIserver.js  Downloads  Pictures   Timeserver.js  node.js.save.1
Desktop      Music      Public     Videos        snap
Documents    Node.js    Templates  node.js.save   time_server.js
ubuntu@localhost:~$ node APIserver.js
```

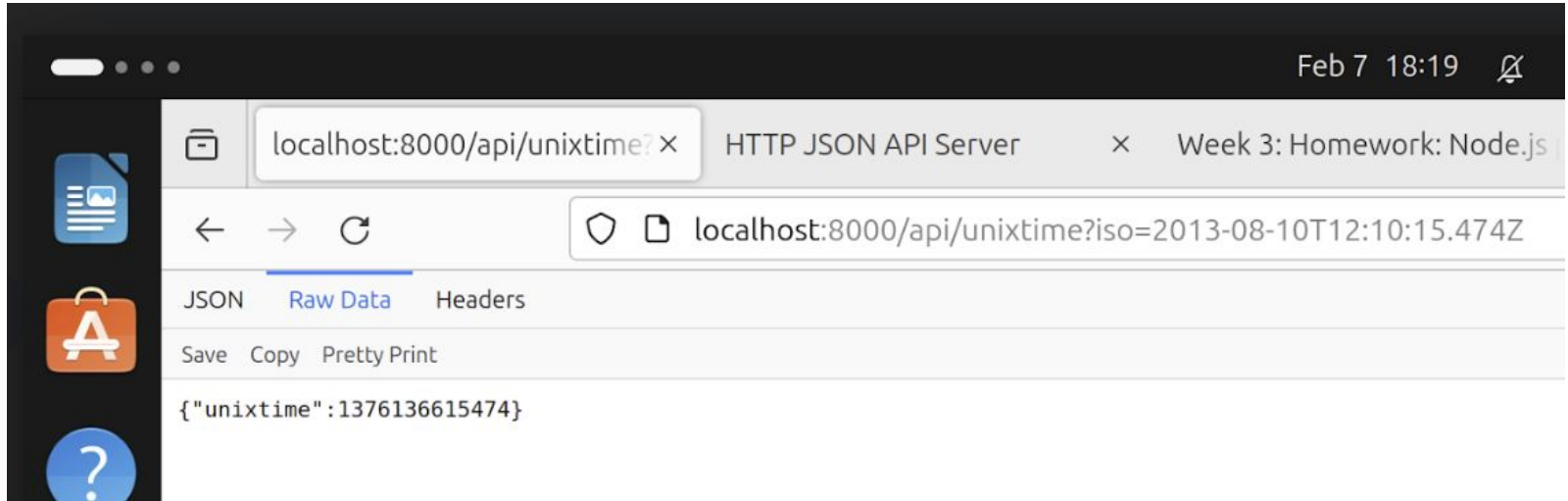
4.3. Type this command into the browser <http://localhost:8000/api/parsetime?iso=2013-08-10T12:10:15.474Z>

- Then current hour, minute and second will be displayed.



4.3. Type this command into the browser <http://localhost:8000/api/unixtime?iso=2013-08-10T12:10:15.474Z>

- Then the unixtime key holding a long integer value representing the number of milliseconds will be displayed.



Step 5: Modify HTTP JSON API Server to support this request from the client side:

5.1. Open text editor in ubuntu and paste the javascript. Then save it as Timeserver.js

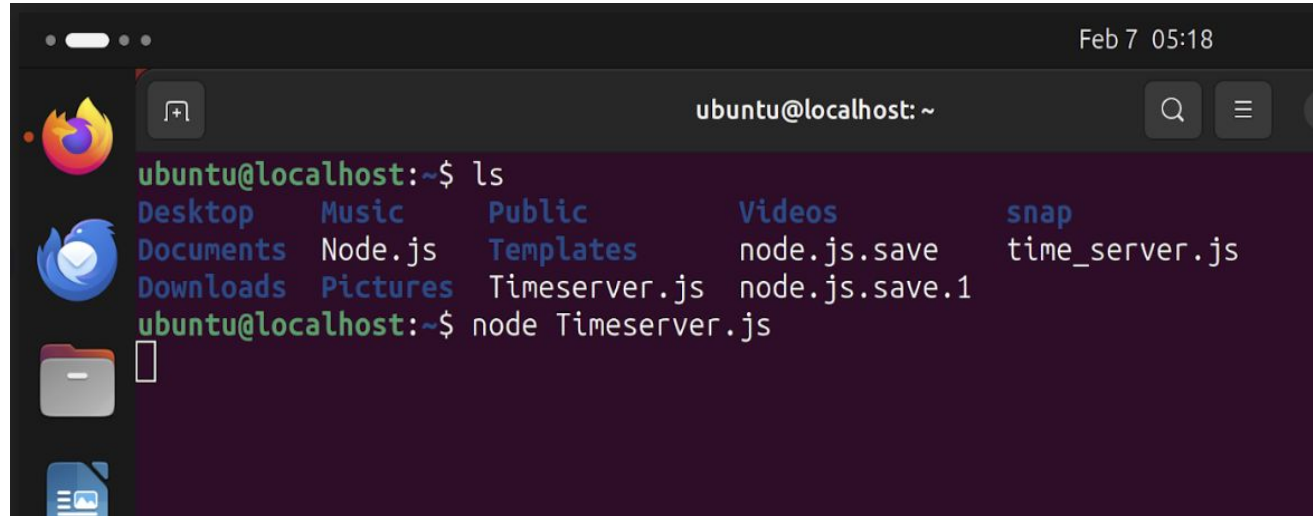
```
var http = require('http');
var url = require('url');

var server = http.createServer(function (req, res) {
  var parsedUrl = url.parse(req.url, true); // Parse the request URL

  if (parsedUrl.pathname === '/api/parsetime') {
    // Handle /api/parsetime endpoint
    if (parsedUrl.query && parsedUrl.query.iso) {
      var time = new Date(parsedUrl.query.iso);
      var response = {
        hour: time.getHours(),
        minute: time.getMinutes(),
        second: time.getSeconds()
      };
      // Set Content-Type header for JSON response
      res.writeHead(200, { 'Content-Type': 'application/json' });
      // Send the JSON response
      res.end(JSON.stringify(response));
    } else {
      // If query string or iso parameter is missing, send 400 Bad Request
      res.writeHead(400);
      res.end();
    }
  } else if (parsedUrl.pathname === '/api/unixtime') {
    // Handle /api/unixtime endpoint
    if (parsedUrl.query && parsedUrl.query.iso) {
      var time = new Date(parsedUrl.query.iso);
      var response = {
        unixtime: time.getTime() // Return UNIX epoch time
      };
      // Set Content-Type header for JSON response
      res.writeHead(200, { 'Content-Type': 'application/json' });
      // Send the JSON response
      res.end(JSON.stringify(response));
    } else {
      // If query string or iso parameter is missing, send 400 Bad Request
      res.writeHead(400);
      res.end();
    }
  } else if (parsedUrl.pathname === '/api/currenttime') {
    // Generate the current date and time
    var currentTime = new Date();
    var response = {
      year: currentTime.getFullYear(),
      month: currentTime.getMonth() + 1, // Adding 1 because getMonth() returns zero-based index
      date: currentTime.getDate(),
      hour: currentTime.getHours(),
      minute: currentTime.getMinutes()
    };
    // Set Content-Type header for JSON response
    res.writeHead(200, { 'Content-Type': 'application/json' });
    // Send the JSON response
    res.end(JSON.stringify(response));
  } else {
    // Handle other endpoints or invalid requests with 404 Not Found
    res.writeHead(404);
    res.end();
  }
});

server.listen(8000); // Listen on port 8000
```

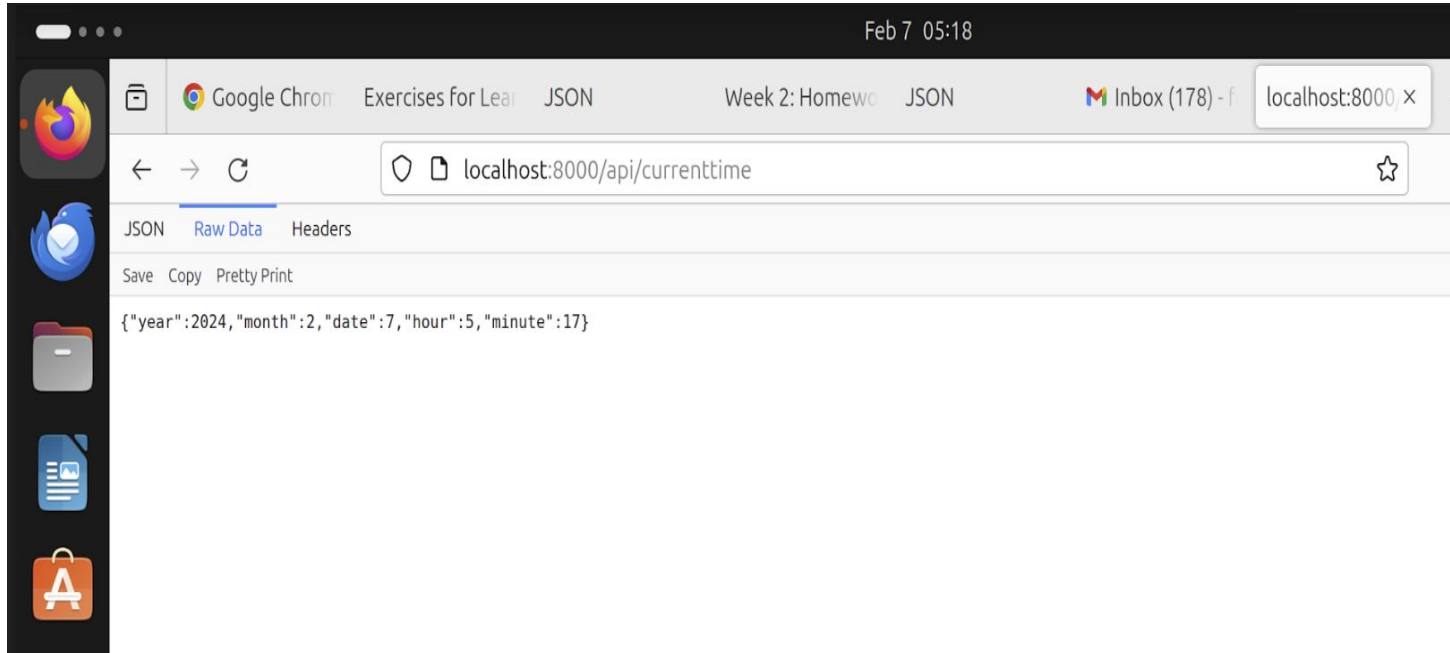
5.2. Open Terminal and then navigate to the previously saved file. Type the command `node Node.js`.

A screenshot of a terminal window on a Linux system. The window title is 'ubuntu@localhost: ~'. The terminal shows the output of the 'ls' command, listing files and directories in the home directory. The files listed are Desktop, Music, Public, Videos, snap, Documents, Node.js, Templates, node.js.save, time_server.js, Downloads, Pictures, Timeserver.js, and node.js.save.1. The user then enters the command 'node Timeserver.js' and the cursor is positioned at the end of the command, waiting for execution.

```
ubuntu@localhost:~$ ls
Desktop  Music    Public   Videos  snap
Documents Node.js  Templates node.js.save time_server.js
Downloads Pictures Timeserver.js node.js.save.1
ubuntu@localhost:~$ node Timeserver.js
```

5.3. Type this command into the browser <http://localhost:8000/api/currenttime>

- Then current year, month,date, hour and minute will be displayed.



4. Enhancement Ideas

- Implementing HTTPS to secure data transmission, adding API authentication to restrict access, and supporting additional time data formats.
- Consideration of extending the API's functionality to include more complex time-related services, such as time conversion across different time zones.

5. Conclusion

- The HTTP JSON API Node.js Time Server project successfully demonstrates a scalable and efficient solution for providing accurate time data to client applications via a simple and accessible API.
- By leveraging Node.js and JSON, the server ensures high performance under concurrent requests and offers a foundation for future enhancements to meet evolving needs.
- This project not only addresses the immediate requirement for reliable time synchronization but also sets the stage for continued innovation in time-serving technologies.

6. References

[HTTP JSON API Server](#)

[JSON](#)

[Node.js on Ubuntu](#)

[Time Server](#)

[Ubuntu Asahi](#)