

CMPT 431 - Assignment 2

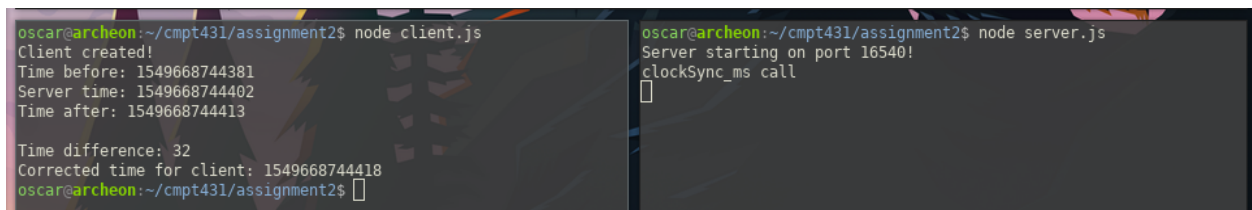
Mayanka Medhe, Oscar Smith-Sieger

February 8, 2019

1

For question 1, we used Nodejs for both the client and server side of things. This was powered by the [node-soap](#) npm packaged for both ends. The WSDL was generated from a [free online tool](#). The client simply stores a timestamp from before the request is made, then makes a request to the **clockSync_ms** remote function, then takes another timestamp from after the request is complete. It uses the before and after timestamps to compute the $RTT/2$, which it then adds to the server time to generate the corrected client time.

The output is:



```
oscar@archeon:~/cmpt431/assignment2$ node client.js
Client created!
Time before: 1549668744381
Server time: 1549668744402
Time after: 1549668744413

Time difference: 32
Corrected time for client: 1549668744418
oscar@archeon:~/cmpt431/assignment2$

oscar@archeon:~/cmpt431/assignment2$ node server.js
Server starting on port 16540!
clockSync_ms call
█
```

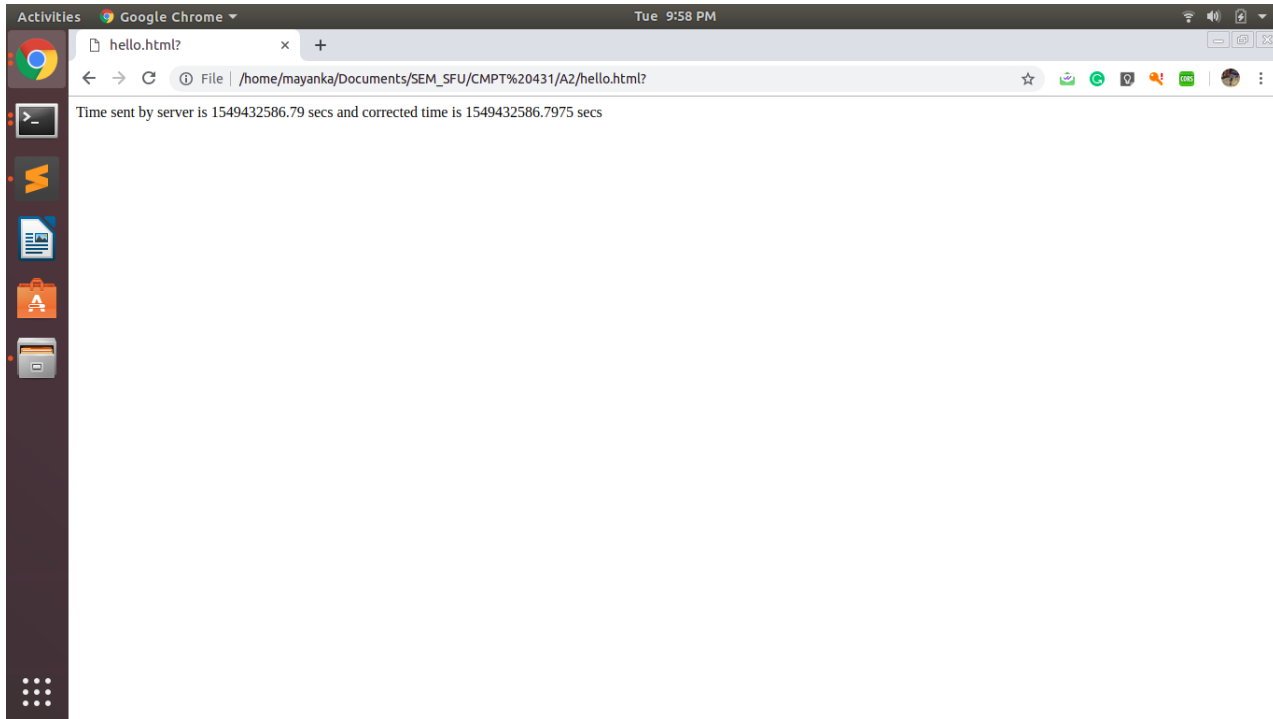
2

In this question, I used **Flask web framework** to deploy my web service. The server is sending the current time using the python **time.time()** function, which returns the time as a floating point number expressed in seconds since the epoch, in UTC. To start the server use command-

```
python q2.py
```

For the client, I have written a HTML file (q2.html), which makes an **XMLHttpRequest** to get the server time and adds half the total time elapsed in receiving the data. In order to successfully run the HTML file, you may need to allow CORS on your Chrome browser. You can add the extension using the following [link](#).

The output is as follows-



3

In this question, I am using python to send API requests to Google calendars to get the upcoming events and to add a new event. The python script uses the credential.json file to get the credentials and stores them in token.pickle for the next run. It accesses the calendar using build() function and retrieves all the upcoming events using .events().list().execute() function with proper arguments. To insert a new event it uses .events.insert() function with proper arguments. To run the script use command -

```
python q3.py
```

The output is as follows-

```
mayanka@mayanka-Aspire-E5-522G:~/Documents/SEM_SFUCMPT 431/A2/q3-2$ python quickstart.py
Getting the upcoming events
Time          Summary          Location
2019-02-07T19:00:00+05:30 Sample event 1 Burnaby, BC, Canada
2019-02-08T19:30:00+05:30 Sample event 2 Burnaby, BC, Canada
2019-02-09T16:30:00+05:30 Sample event 3 Mumbai, India
2019-02-11T15:30:00+05:30 Sample event 4 Simon Fraser University, Burnaby
2019-02-13T15:30:00+05:30 Sample event 5 Metrotown
Event created: https://www.google.com/calendar/event?eid=MWRwaXVnczFLM2VnaTAwMnRsYXJkYm1nMzggbWFSYk5rYWVuaWxtZWRoZUBt
mayanka@mayanka-Aspire-E5-522G:~/Documents/SEM_SFUCMPT 431/A2/q3-2$
```

The screenshot displays the Google Calendar web application running in a Chromium browser on a Linux desktop. The desktop environment includes a sidebar with application icons for Activities, Google Chrome, and various system utilities. The browser window shows the Google Calendar interface for February 2019. The calendar is set to 'Week' view, and the current date is February 7, 2019. A meeting titled 'Meeting' is scheduled for February 6, 2019, from 10:30 PM to 6:30 AM at Simon Fraser University, Burnaby, Canada. The meeting is presented by Prof. Michael Collins and is set to notify attendees 10 minutes before and 1 day before via email. The calendar also shows a 'Happy birthday!' notification for February 8, 2019. The sidebar on the left lists 'My calendars' including 'Mayanka Medhe', 'Birthdays', 'Reminders', and 'Tasks', and 'Other calendars' including 'Holidays in India'.