Task 1: Implement a Web Analytics Service

Last week, you implemented a small web server. In this task, the server's functionality will be extended to process the following requests:

- A request to "/" replies with the current time and date. Furthermore, the number of calls the server already served should be displayed.
  - Hint: you need to store it on the server. Restarting the server should **not** reset the counter.
- A request to "/whoami" replies with as much information as you can get from the requests of the user. This can, for example, include:
  - Location (coordinates, place, …)
  - The web browser, installed plugins, …
  - Connection metadata: IP address, download speed, …
  - Optional for mobile devices: Sensor data (Gyroscope, accelerometer, …)
- A request to "/adapt2user" provides a reply that is different depending on who is requesting the page. The minimum requirement is to implement at least 3 of the following:
  - Different pages for inside and outside the LMU network
  - Different pages for different browsers
  - Different pages for different operating systems
  - Different pages for first time access (only using IP address – no cookies)
  - Different pages for mobile access
  - Different pages for different language settings

If you want, you can also create the server using Node.js. You will need to install Node.js on your platform (https://nodejs.org/en/download/). Basic introductions can be found here:
https://nodejs.org/en/docs/guides/getting-started-guide/
https://www.tutorialspoint.com/nodejs/nodejs_first_application.htm
The Node.js package manager npm comes with a variety of packages to accomplish the tasks. Some examples are:
https://www.npmjs.com/package/detect-browser
https://www.npmjs.com/package/useragent

For inspiration, here is a (creepy) example of what you could find out about a client issuing a web request:
http://webkay.robinlinus.com/

Task 2: Web Analytics

In this task, you should describe what web analytics are. Research the web about the definition of web analytics and understand the concept behind it. Write a summary of approximately 200 words. This summary should include:

- A definition of web analytics
- Common packages and services which are used for web analytics

- A description of how web analytics work
- A discussion about the legal and ethical perspectives of web analytics

Please note that you should include links and references of your web research.

Task 3: Chat Server Concept and Architecture

Develop a concept and an architecture for a basic chat server. The chat server uses HTTP as protocol to communicate content sentence by sentence:

- Describe a communication system that uses HTTP to implement a chat server. You can use graphical (e.g. UML (https://www.smartdraw.com/uml-diagram/)) visualization and textual descriptions to describe such a system. Explain the basic architecture behind such a system.
- Optional: Think about how such a system can be designed to hide an ongoing conversation i.e., a person monitoring HTTP traffic is not able to identify a chat. It should look like regular HTTP requests which are send to a server. Think about the information send by the browser (Task 1) and HTTP-Headers. Provide a graphical or textual description of how this could be accomplished.

Additional Material: JavaScript Tutorial and e-Books

- JavaScript Tutorial: https://www.w3schools.com/js/default.asp
- Axel Rauschmayer. Speaking JavaScript: An In-Depth Guide for Programmers, http://speakingjs.com/es5/index.html
- Marijn Haverbeke, Eloquent JavaScript, 3rd edition, http://eloquentjavascript.net http://eloquentjavascript.net/Eloquent_JavaScript.pdf

Modalities for handing in your results:

Please hand in a zip file of your source code (**Deadline: 04.11.18, 23:59**). Upload a zip file to Uniworx with the following format: StudentID_Lastname_Firstname (German: Matrikelnummer_Nachname_Vorname). The zip file should contain the following file structure:

- StudentID_Lastname_Firstname
  - Task 1
    - Commented and referenced source code
  - Task 2
    - Your summary
  - Task 3
    - The communication architecture and (optionally) a concept of how an ongoing chat can be hidden

For example, a student with the name Joe Doe has the student number 123456. The file would look this way: “123456_Joe_Doe.zip”.