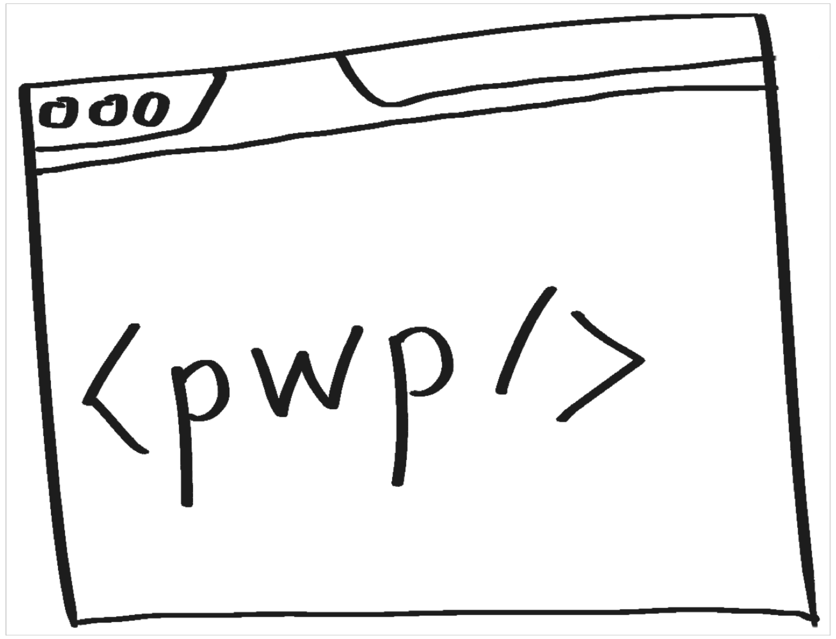
Practical Course: Web Programming

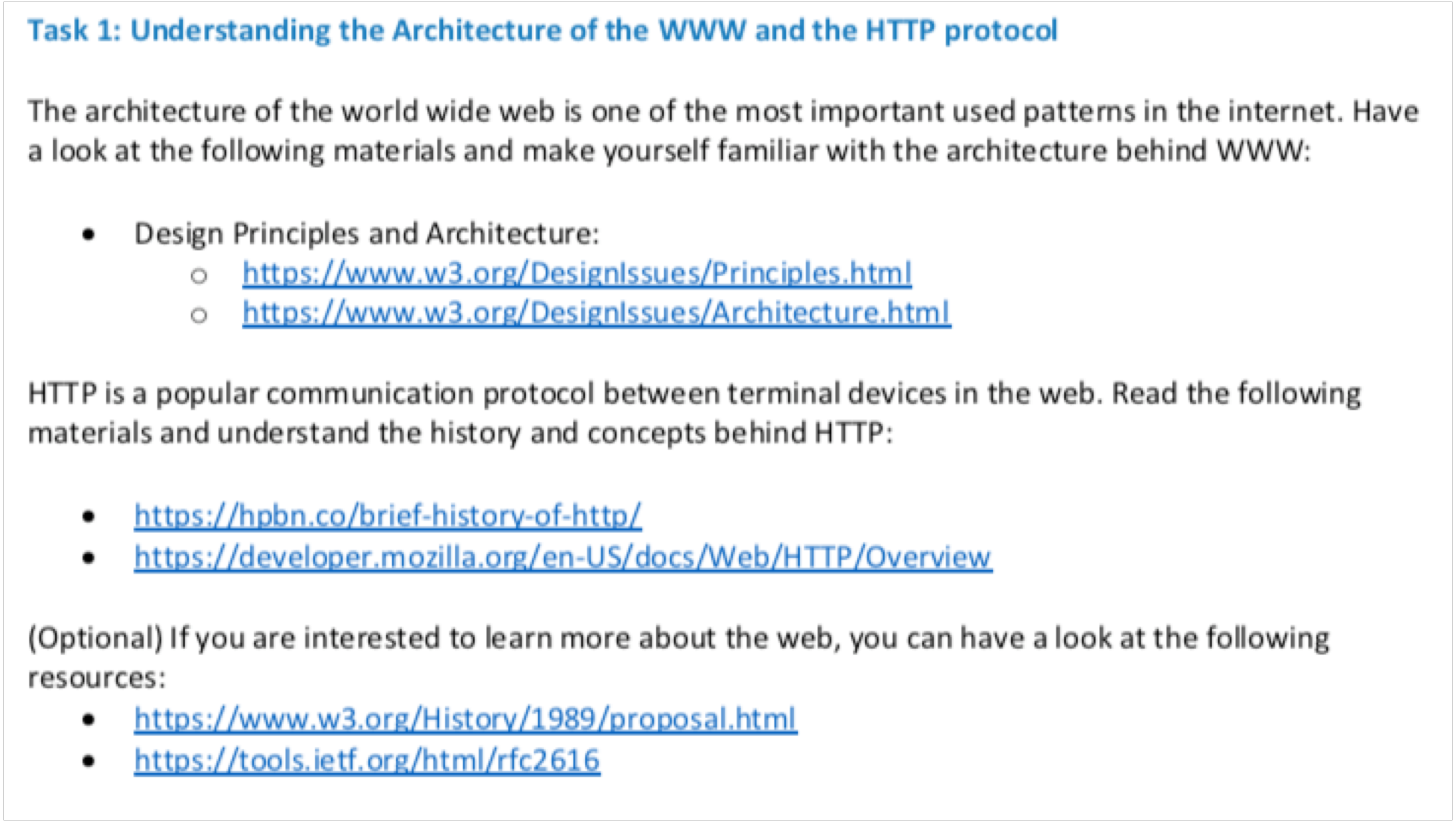
Human-Centered Ubiquitous Media
Prof. Dr. Albrecht Schmidt, Fiona Draxler,
Thomas Kosch

Winter Term 18/19



Addendum: Course Organisation

- Ubiaction at the end of the semester
- Event where participants of seminar and practicals present their results
- Prepare a poster/demo of your group projects



Task 1: Understanding the Architecture of the WWW and the HTTP protocol

The architecture of the world wide web is one of the most important used patterns in the internet. Have a look at the following materials and make yourself familiar with the architecture behind WWW:

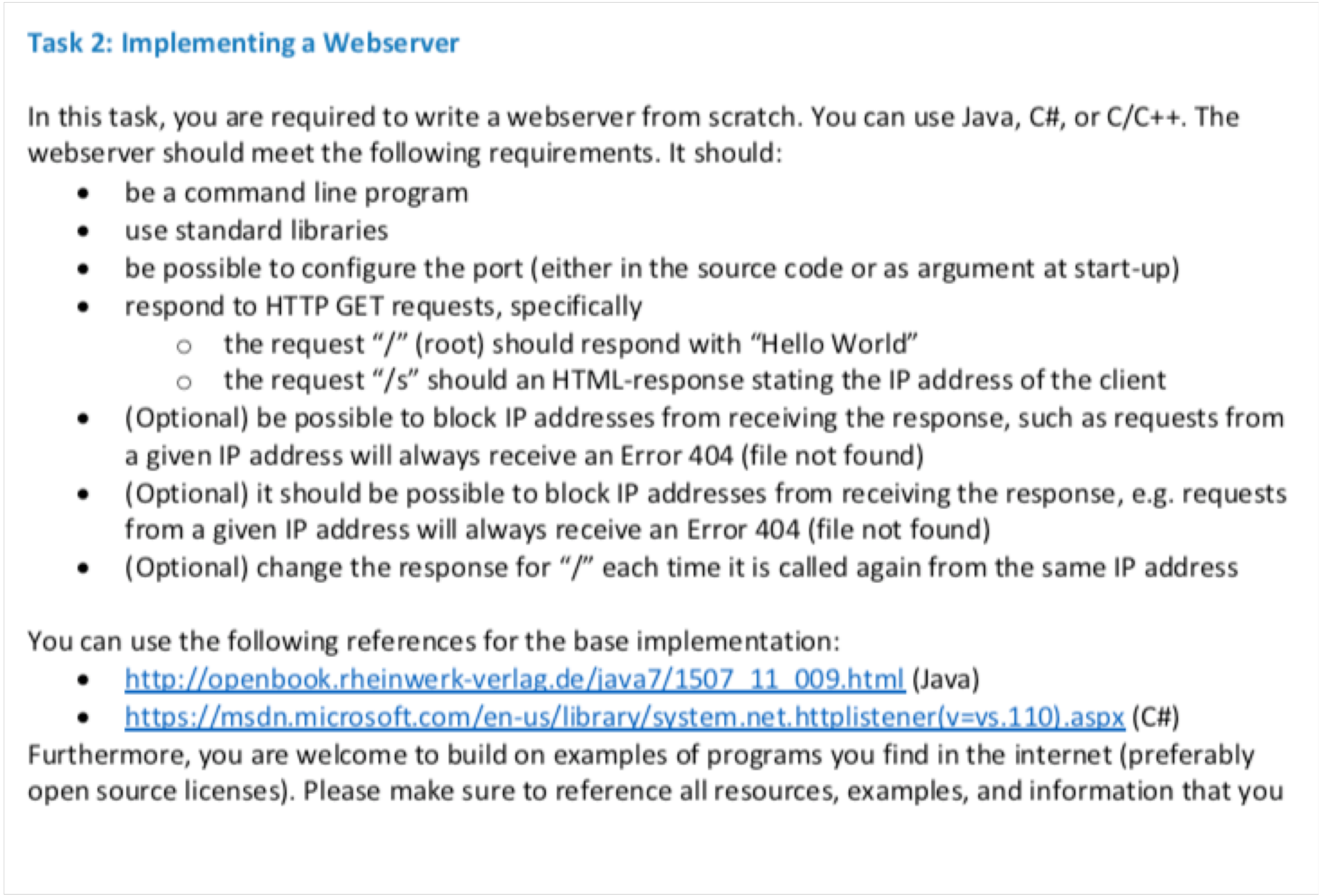
- Design Principles and Architecture:
    - https://www.w3.org/DesignIssues/Principles.html
    - https://www.w3.org/DesignIssues/Architecture.html

HTTP is a popular communication protocol between terminal devices in the web. Read the following materials and understand the history and concepts behind HTTP:

- https://hpbn.co/brief-history-of-http/
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

(Optional) If you are interested to learn more about the web, you can have a look at the following resources:

- https://www.w3.org/History/1989/proposal.html
- https://tools.ietf.org/html/rfc2616



Task 2: Implementing a Webserver

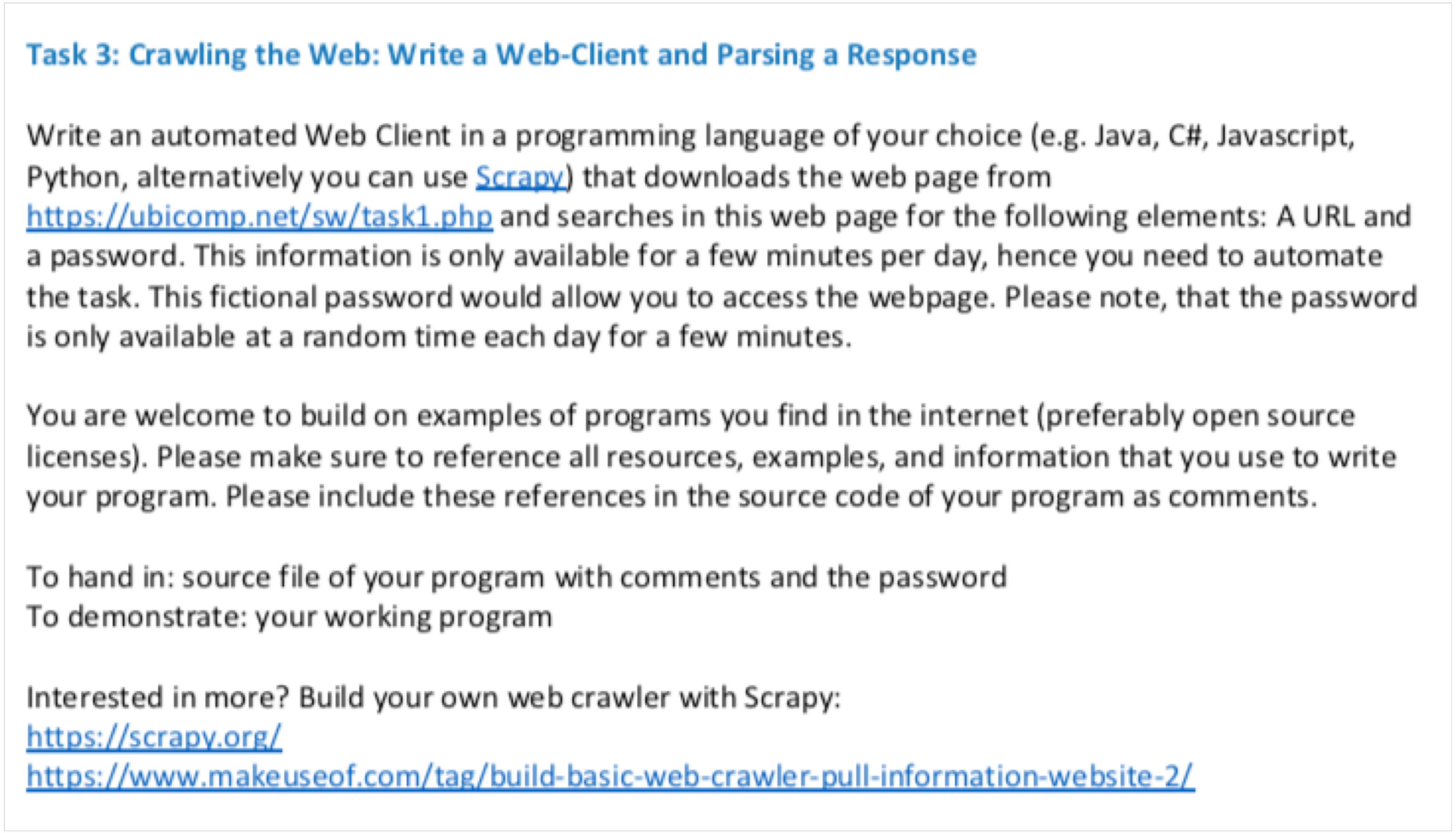
In this task, you are required to write a webserver from scratch. You can use Java, C#, or C/C++. The webserver should meet the following requirements. It should:

- be a command line program
- use standard libraries
- be possible to configure the port (either in the source code or as argument at start-up)
- respond to HTTP GET requests, specifically
  - the request "/" (root) should respond with "Hello World"
  - the request "/s" should an HTML-response stating the IP address of the client
- (Optional) be possible to block IP addresses from receiving the response, such as requests from a given IP address will always receive an Error 404 (file not found)
- (Optional) it should be possible to block IP addresses from receiving the response, e.g. requests from a given IP address will always receive an Error 404 (file not found)
- (Optional) change the response for "/" each time it is called again from the same IP address

You can use the following references for the base implementation:

- http://openbook.rheinwerk-verlag.de/java7/1507_11_009.html (Java)
- https://msdn.microsoft.com/en-us/library/system.net.httplistener(v=vs.110).aspx (C#)

Furthermore, you are welcome to build on examples of programs you find in the internet (preferably open source licenses). Please make sure to reference all resources, examples, and information that you



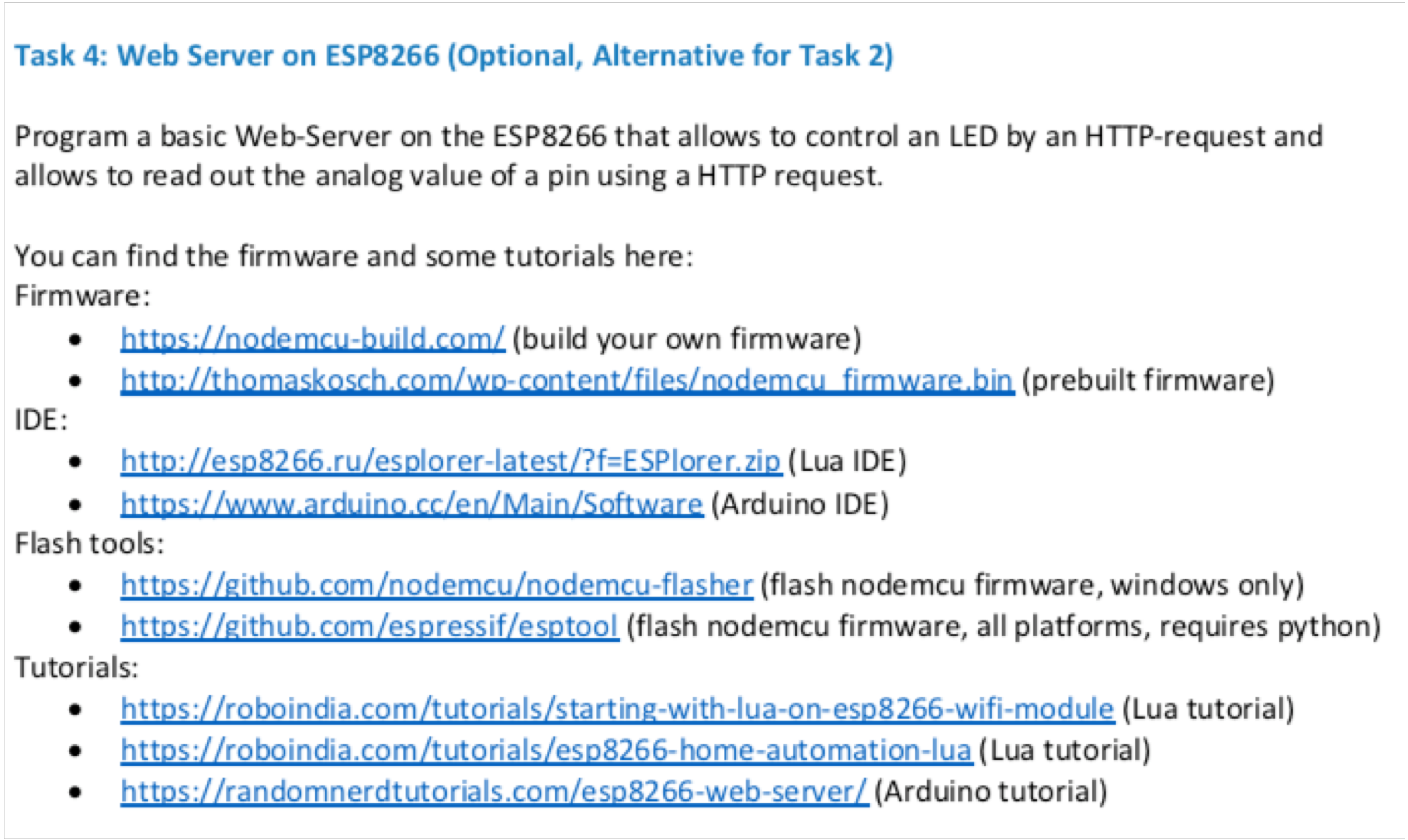
Task 3: Crawling the Web: Write a Web-Client and Parsing a Response

Write an automated Web Client in a programming language of your choice (e.g. Java, C#, Javascript, Python, alternatively you can use Scrapy) that downloads the web page from https://ubicomp.net/sw/task1.php and searches in this web page for the following elements: A URL and a password. This information is only available for a few minutes per day, hence you need to automate the task. This fictional password would allow you to access the webpage. Please note, that the password is only available at a random time each day for a few minutes.

You are welcome to build on examples of programs you find in the internet (preferably open source licenses). Please make sure to reference all resources, examples, and information that you use to write your program. Please include these references in the source code of your program as comments.

To hand in: source file of your program with comments and the password
To demonstrate: your working program

Interested in more? Build your own web crawler with Scrapy:
https://scrapy.org/
https://www.makeuseof.com/tag/build-basic-web-crawler-pull-information-website-2/



Task 4: Web Server on ESP8266 (Optional, Alternative for Task 2)

Program a basic Web-Server on the ESP8266 that allows to control an LED by an HTTP-request and allows to read out the analog value of a pin using a HTTP request.

You can find the firmware and some tutorials here:

Firmware:

- https://nodemcu-build.com/ (build your own firmware)
- http://thomaskosch.com/wp-content/files/nodemcu_firmware.bin (prebuilt firmware)

IDE:

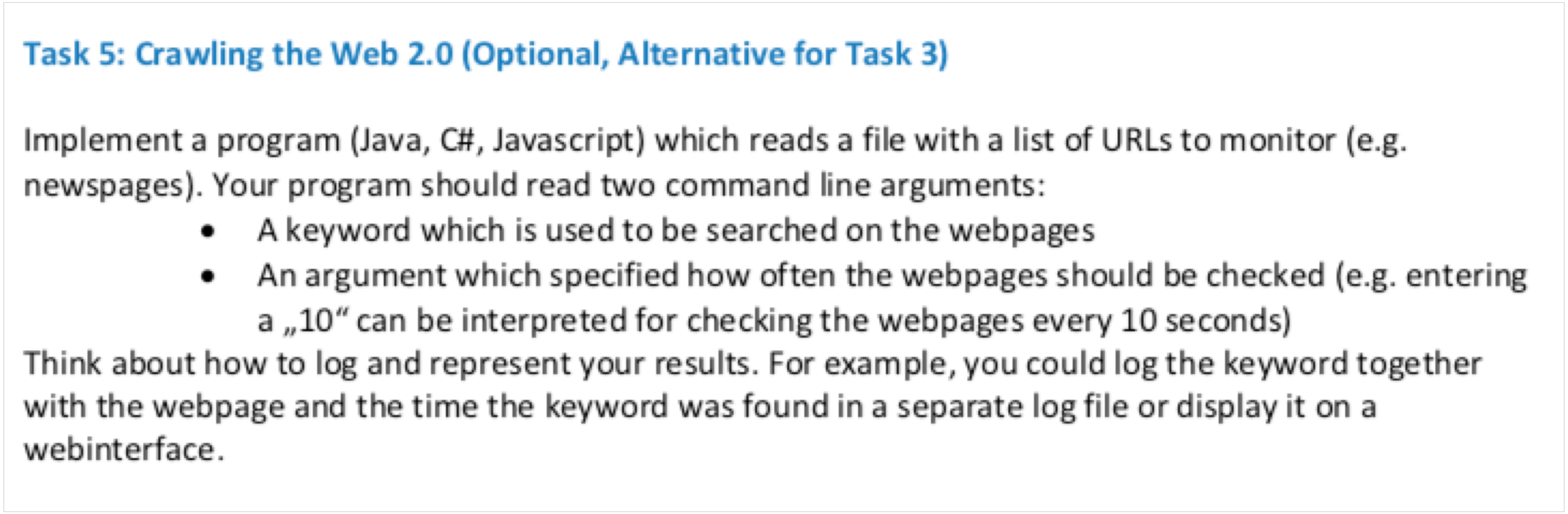
- http://esp8266.ru/esplorer-latest/?f=ESPlorer.zip (Lua IDE)
- https://www.arduino.cc/en/Main/Software (Arduino IDE)

Flash tools:

- https://github.com/nodemcu/nodemcu-flasher (flash nodemcu firmware, windows only)
- https://github.com/espressif/esptool (flash nodemcu firmware, all platforms, requires python)

Tutorials:

- https://roboindia.com/tutorials/starting-with-lua-on-esp8266-wifi-module (Lua tutorial)
- https://roboindia.com/tutorials/esp8266-home-automation-lua (Lua tutorial)
- https://randomnerdtutorials.com/esp8266-web-server/ (Arduino tutorial)



Task 5: Crawling the Web 2.0 (Optional, Alternative for Task 3)

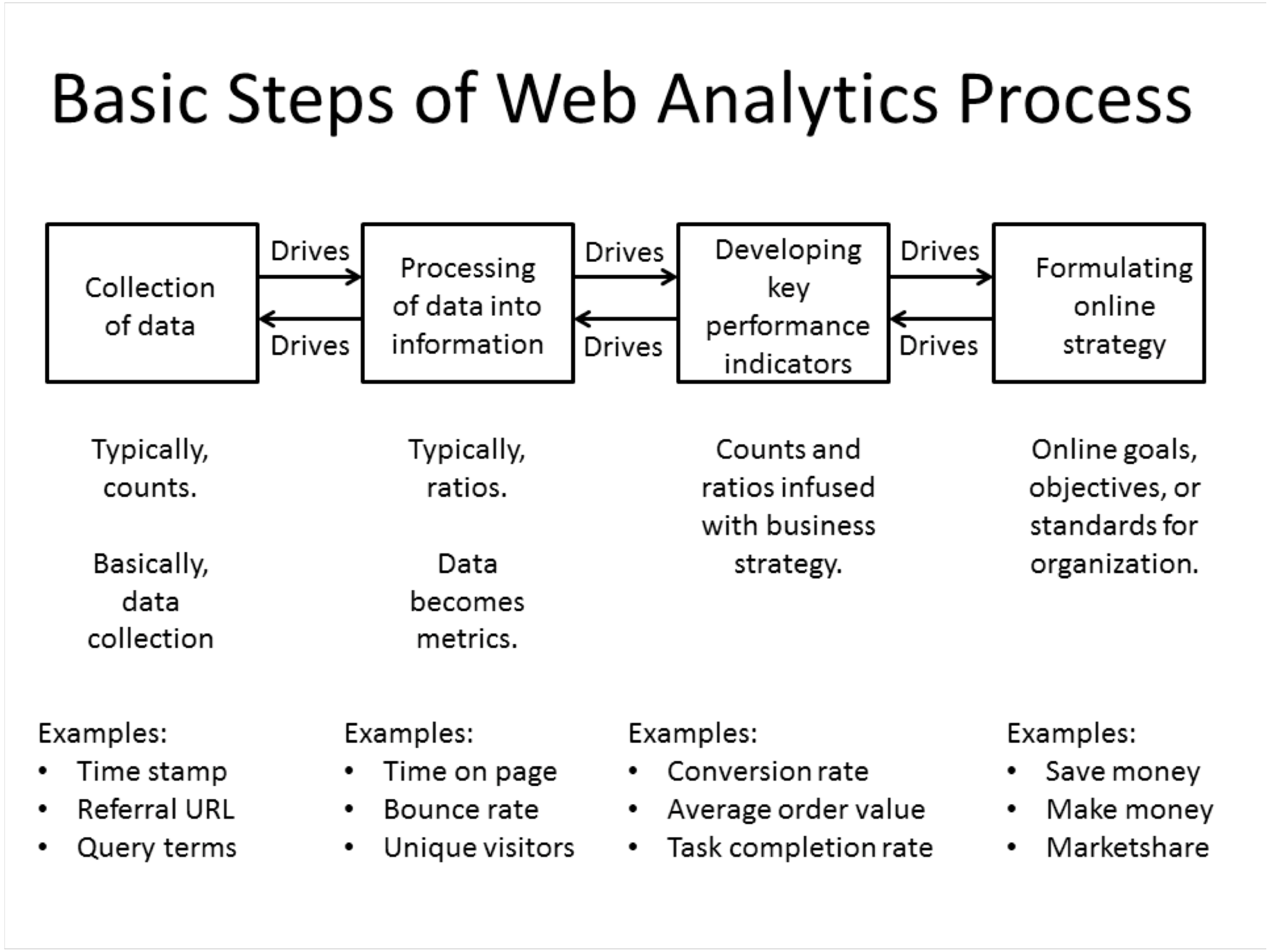
Implement a program (Java, C#, Javascript) which reads a file with a list of URLs to monitor (e.g. newspages). Your program should read two command line arguments:

- A keyword which is used to be searched on the webpages
- An argument which specified how often the webpages should be checked (e.g. entering a „10“ can be interpreted for checking the webpages every 10 seconds)

Think about how to log and represent your results. For example, you could log the keyword together with the webpage and the time the keyword was found in a separate log file or display it on a webinterface.

What are web analytics?

How is data collected?


Collection of data
Drives
Drives
Processing of data into information
Drives
Drives
Developing key performance indicators
Drives
Drives
Formulating online strategy

Basic Steps of Web Analytics Process

Collection of data: Typically, counts. Basically, data collection.

Examples:
- Time stamp
- Referral URL
- Query terms

Processing of data into information: Typically, ratios. Data becomes metrics.

Examples:
- Time on page
- Bounce rate
- Unique visitors

Developing key performance indicators: Counts and ratios infused with business strategy.

Examples:
- Conversion rate
- Average order value
- Task completion rate

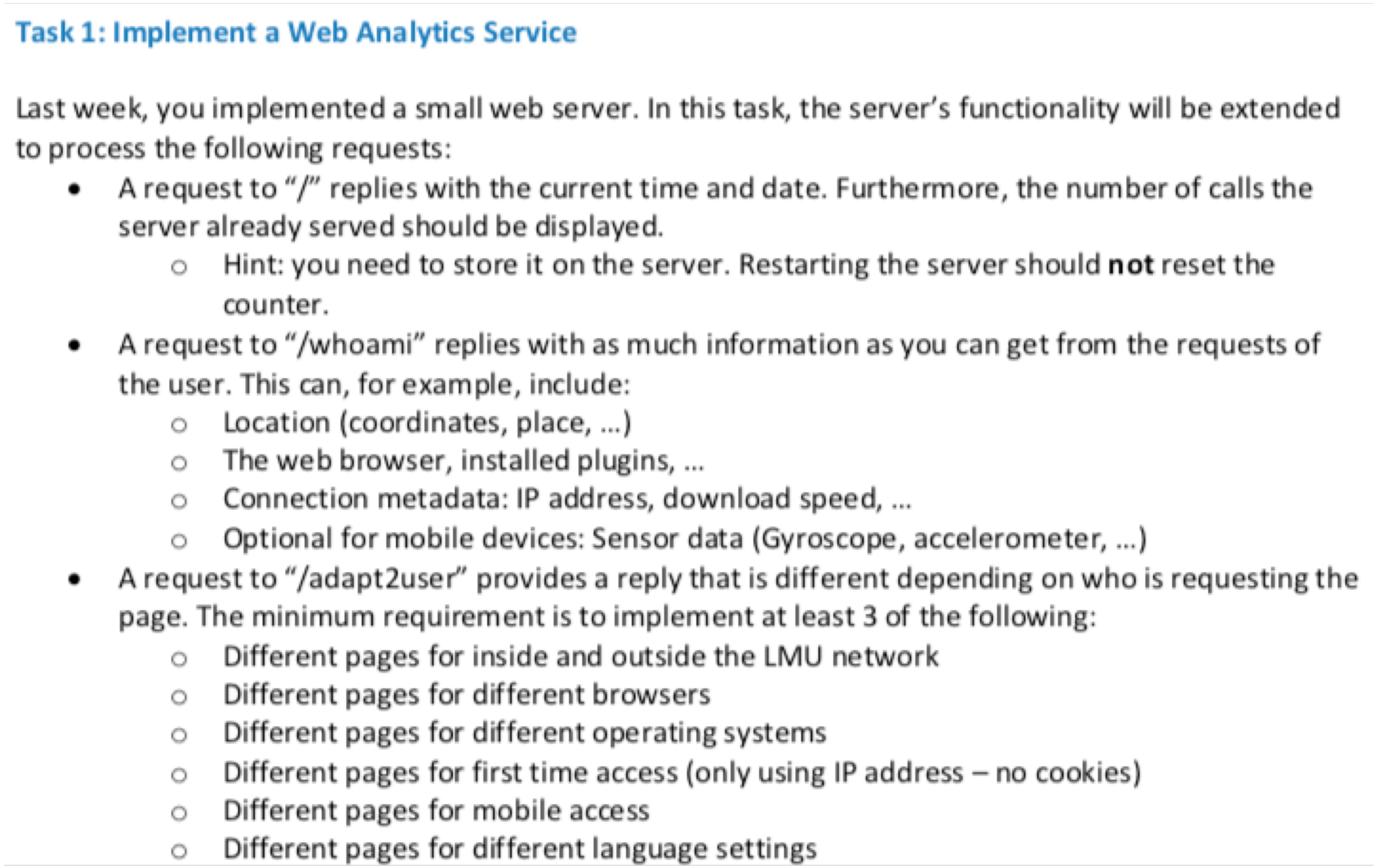
Formulating online strategy: Online goals, objectives, or standards for organization.

Examples:
- Save money
- Make money
- Marketshare

Image from: https://upload.wikimedia.org/wikipedia/commons/f/f1/Basic_Steps_of_Web_Analytics_Process.png

Web Analytics, Privacy, and Security

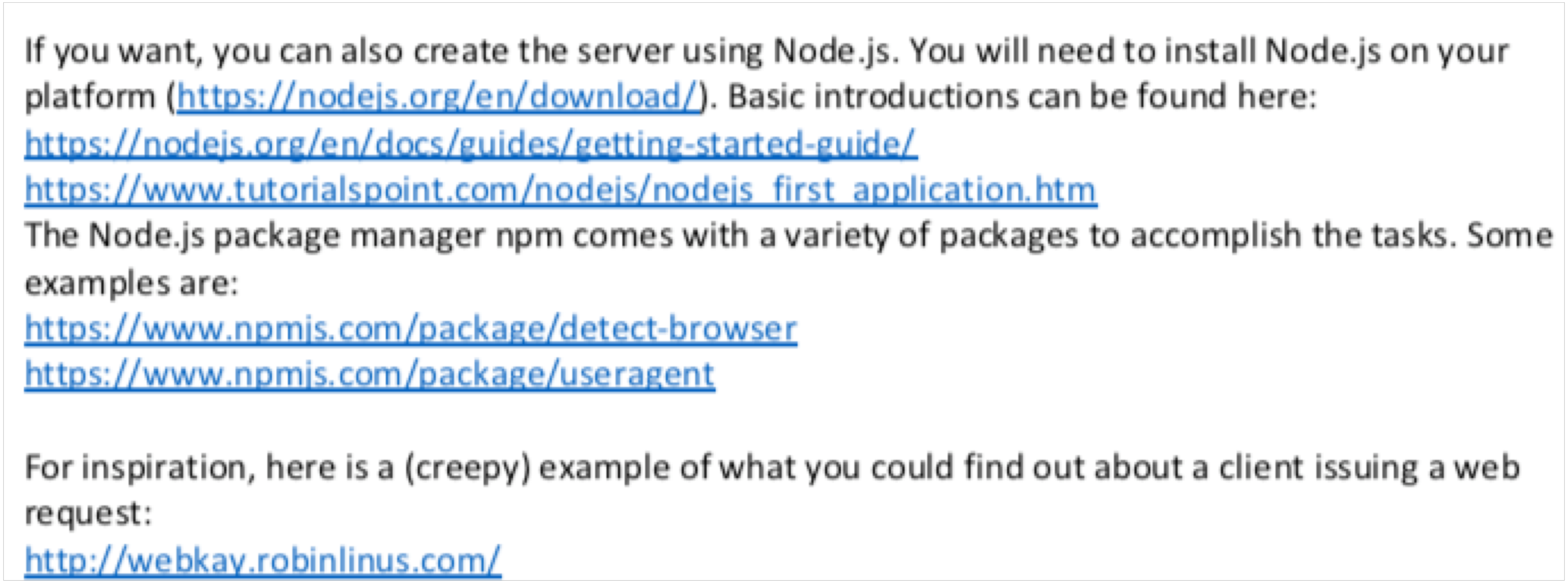
- What are cookies? What were they originally intended for?
- What other tracking methods except cookies are there?
- What can you find out about someone accessing a website?
- What does “do not track“ mean?
- What difference does private browsing make?
- What does the GDPR (DSGVO) have to do with all this?
- What is the difference between HTTP and HTTPS?



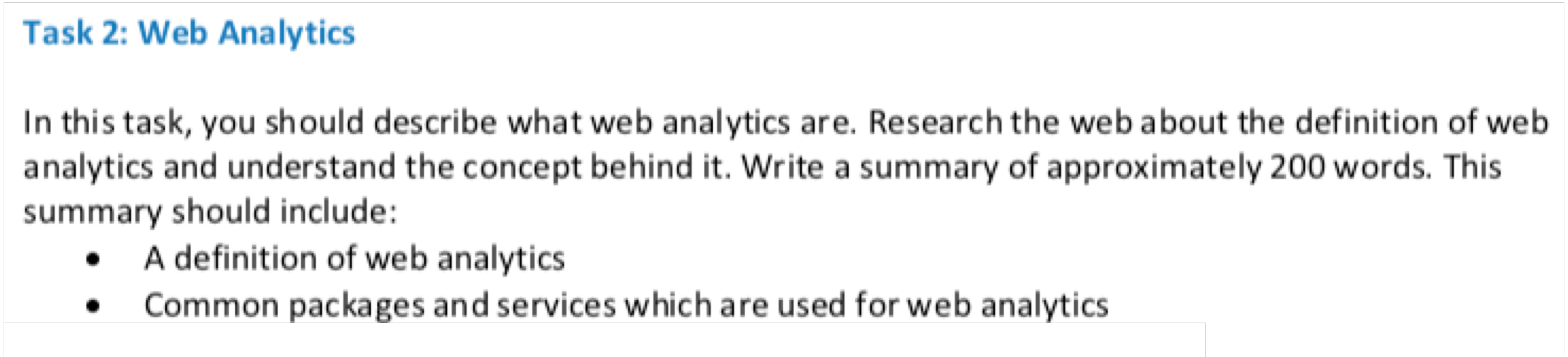
Task 1: Implement a Web Analytics Service

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

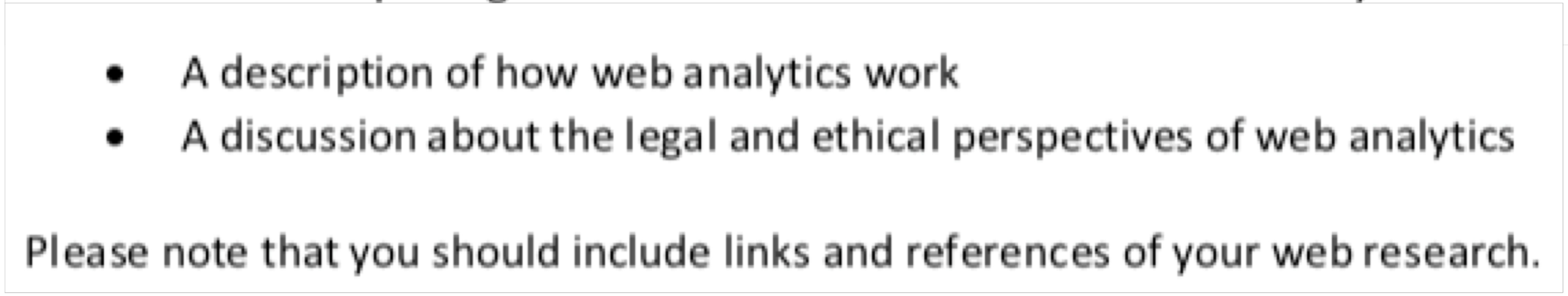
For inspiration, here is a (creepy) example of what you could find out about a client issuing a web request:
http://webkay.robinlinus.com/



Task 2: Web Analytics

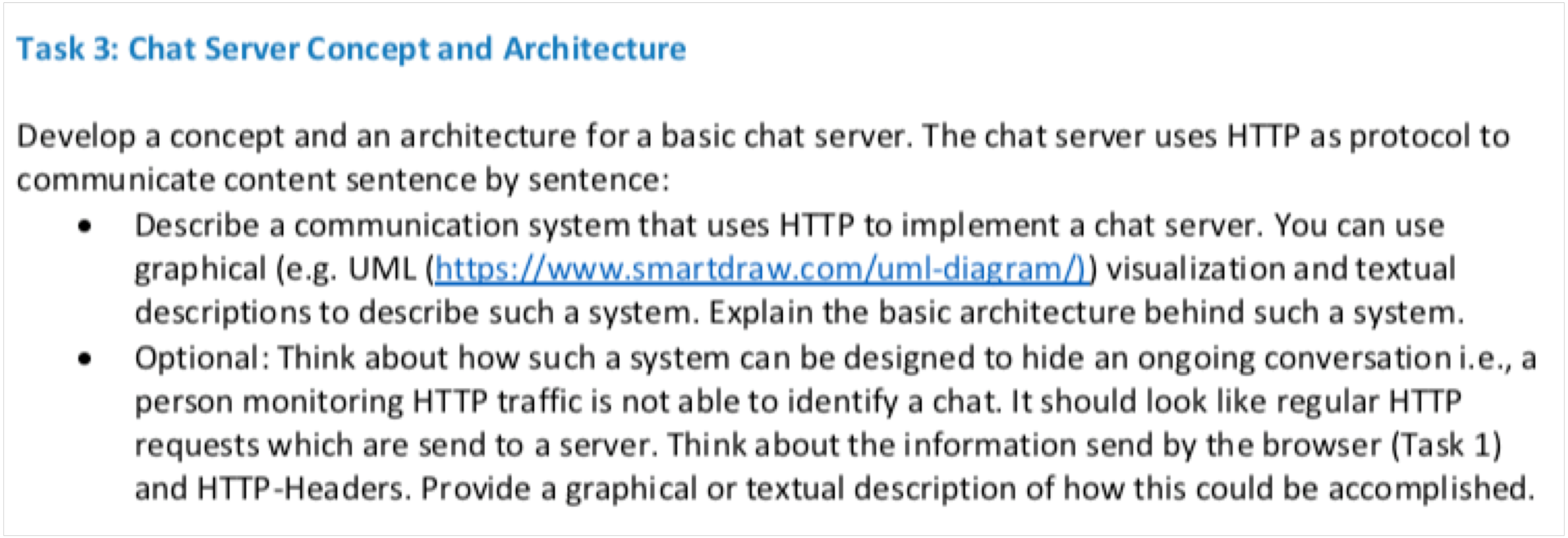
In this task, you should describe what web analytics are. Research the web about the definition of web analytics and understand the concept behind it. Write a summary of approximately 200 words. This summary should include:

- 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.