IMY 220
Assignment 9:
node.js

Due: Wednesday 2 October @ 13:00

The submission instructions are available on ClickUP. Your assignment will not be marked if you do not follow these instructions.

Instructions

- The aim of this assignment is to create a server that uses two Node.js modules, Express and Socket.io, to allow users to write a message and which saves this information to a file.
- There are no files to download for this assignment. You must create all of them yourself.

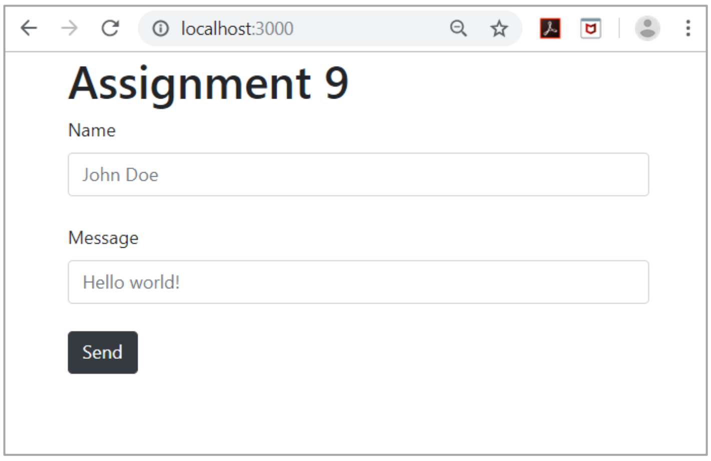
Section 1 – Installing modules

Install Express and Socket.io to your working directory via the command line. The modules must also be added to your ***package.json*** file as dependencies. If this is done correctly, there should be a folder named "*node_modules*" in your working directory which contains two folders named "*express*" and "*socket.io*" and the packages should be listed as dependencies in your *package.json* file. Take note that you must **not** submit the *node_modules* directory along with your assignment, but you must submit your *package.json* file which must contain valid values for at least the "name", "version" and "description" fields (as well as the dependencies for Express and socket.io).

Section 2 – Serve a static webpage

Using Express, serve a webpage that provides a simple form. This should work as follows:

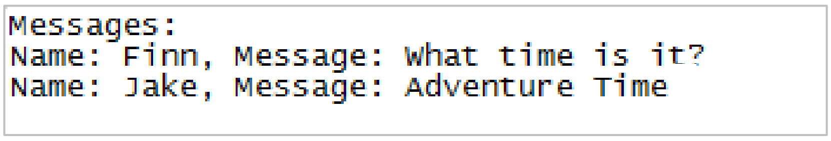
- Create a new file called *index.js*. This file will contain all your node.js server code
- Use Express to serve static content. This content should be found inside a directory called "*public*".
- Inside this directory, create a file called *index.html* which contains a simple Bootstrap form with two text boxes (one for name and one for message) and a submit button.
- The server should listen at localhost:3000.
- This should look something like this:



Section 3 - Send messages to the server and write to file

Using socket.io, add functionality to send the name and message that user entered into the form back to the server and to write it to a file called *messageLog.txt*. This should work as follows:

- When the server starts up, it should create a file called *messageLog.txt* if it doesn't exist and overwrite it if it exists. Upon creating this file, it should contain the text: "Messages:"
- When the user hits "Send" on the input form, the information (name and message) should be read from the form using jQuery and saved as a JSON object.
- This JSON object should be sent back to the server. This should be done using socket.io.
- The server should catch the emitted information and write it to *messageLog.txt* in the following format:
    - Name: <name>, Message: <message>
- Each entry should be on a new line.
- For example:

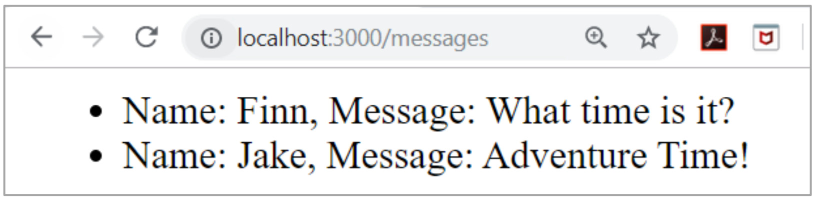


```
Messages:
Name: Finn, Message: What time is it?
Name: Jake, Message: Adventure Time
```

Bonus section – view messages

Create functionality for viewing messages on a separate page. This has to work as follows:

- When a user navigates to *localhost:3000/messages* it should serve different content (i.e. not the form for sending messages)
- This content must a bulleted list of every message that has been sent using the form you created for section 2. For example, if I had sent two messages using the form and opened *localhost:3000/messages* in a separate tab, I might see something like this.



- Name: Finn, Message: What time is it?
- Name: Jake, Message: Adventure Time!

- In order to accomplish this, you will have to change the functionality for saving the messages to a file. Specifically, you will have to use a .json file and save each message in such a way to ensure that the contents of the file is valid JSON at all times.
- HINT: you can use https://jsonlint.com/ to check for valid JSON
- Then, when navigating to *localhost:3000/messages*, you must read the JSON file, parse its contents as a JS object, loop through all the messages inside this file, and display each message as a bulleted list item.

Additional Information

- Refer to the slides and node.js website for help

Submit only the following file(s) according to the submission instructions.

- index.js
- package.json
- “public” directory with the following files
    - index.html