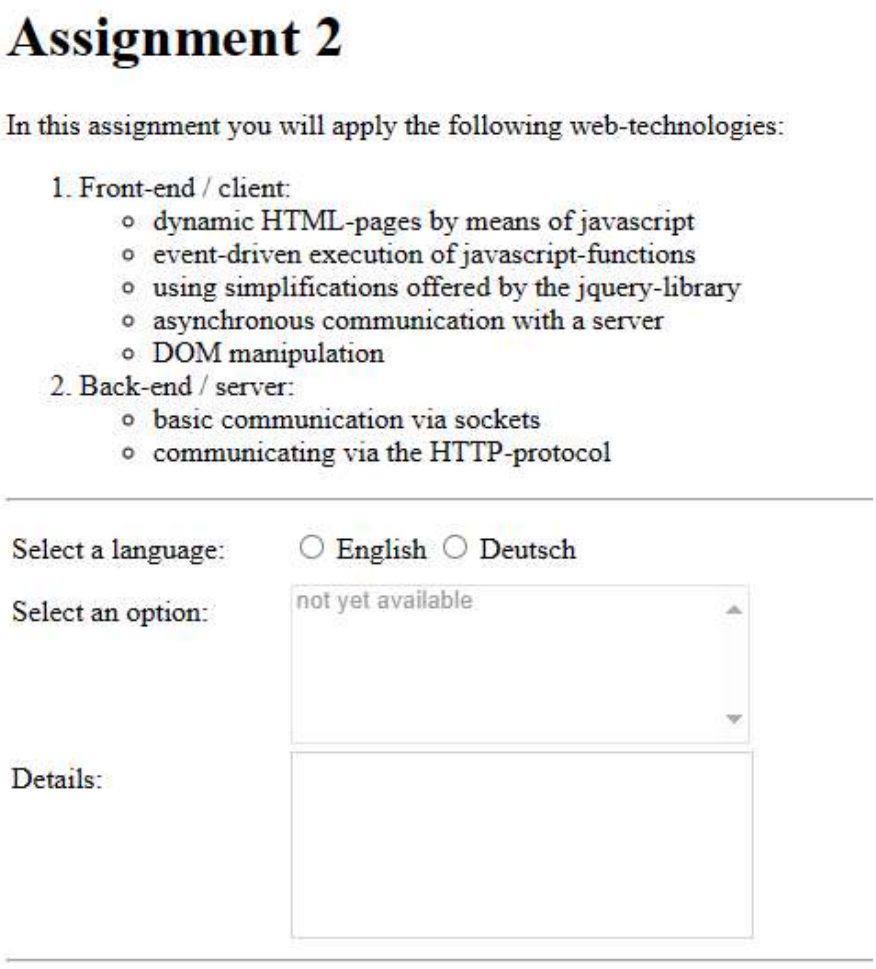
Practical Course "Web Technologies"

Assignment 2

Implement a communication between a browser and a simple server.

1. the client
   a. a simple HTML-page which looks exactly like



   Initially,

   - no button must be selected in the radio-button group
   - the selection list must be disabled

   The details-rectangle is a container/div-element with a border. It is used to display additional text (see below). The elements can differ in width, height, and – slightly – in style. You can use inline-styles.

   b. a javascript-script in a separate file that contains the following functions

   - an event function associated with the radio-buttons: if the radio-button's selection changes, a POST-request is sent asynchronously to the server. A language-specific list of options is requested accordingly. The request contains information about the

selected language. The details area is cleared. When the response is received, the response data is displayed in the selector list.

The list of options is transmitted in json-format as an array of arrays. An option is described by an array/tuple and consists of three values: a numerical identification, a string to be shown as option in the selection list, and additional data. The server defines the options as follows:

```
OPTIONS_DE = list()
OPTIONS_DE.append( (0, 'Zwei Herren aus Verona', '1590/91') )
OPTIONS_DE.append( (1, 'Ein Sommernachstraum', '1595') )
OPTIONS_DE.append( (2, 'Die lustigen Weiber von Windsor', '1597/98') )

OPTIONS_EN = list()
OPTIONS_EN.append( (0, 'The Two Gentlemen of Verona', '1590/91') )
OPTIONS_EN.append( (1, "A Midsummer Night's Dream", '1595') )
OPTIONS_EN.append( (2, 'The Merry Wives of Windsor', '1597/98') )

OPTIONS = {0: OPTIONS_EN, 1: OPTIONS_DE}
```

- an event function associated with the selection list: if an option of the selection list is selected it displays the additional data associated with the selected option in the details area

2. the server

   The server communicates with the browser via sockets. The communication is based on HTTP.

   When started, the server creates a socket and binds the address (127.0.0.1, 5600) to it. Then, it listens for connection requests. The connection request is the very first HTTP-request and is initiated when a browser requests the resource with the URL 'localhost:5600'. You can suffix the URL with the name of your HTML-document. However, if suffixed or not your HTML-document is addressed. The server accepts the connection request. The reference to the socket returned by accept is used for the following dialogue. Note, the very first HTTP-request establishes the connection between server and client as well as requests the HTML-document.

   The server analyzes the HTTP-requests, i.e. it determines the HTTP-method, the URL, and the request's entity. GET-requests are answered by returning the requested resource (e.g. HTML-file, javascript-file). POST-requests are answered by returning the list of options in json-format specified in the POST's entity.

   The servers must prepare the HTTP-responses. A simple header is prepared (see attached py-file). It must be extended by the current time, the length of the content and the content, i.e. either the content of the file addressed by the request or the json-string of the requested list of options.

   The main function of the server consists of: setting up the connection to the client, repeatedly receiving a request, processing the request, returning the response. Structure your server by means of functions accordingly.