Software Construction: Q & A Project Description
(Current State: October 3nd)

(Added October 4th)

- *Question*: Can we use external libraries?
  Answer: No, all functionality has to be designed and implemented by yourself. (Personally, I don't think using external libraries would actually help much for this project.)

- *Question:* How should the output be? In an abstracted manner (e.g. for cli/file)?
  Answer: The library should return the generated output as a string. Because the user of the library can then decide how to output it (e.g., stdout or to a file).

- *Question*: Do we have to specify the API in the SRS?
  Answer: No, this will be the task for the next milestone. The SRS should be specify in a way that you can give it to a different development team and they would be able to write the library in the way the client envisioned it.

- *Question*: Are use cases necessary?
  Answer: You can use them if you find them helpful for your description. However, we do not require them.

- *Question*: Is the deadline midnight?
  Answer: Yes, until 23:59. That's when we will make a snapshot of your repositories.

(Added October 3rd)

- *Question:* I am still not sure how a user provides the input for the resulting HTML document?
  Answer: I think this comes from a misunderstanding of what a software library is. Here a quick excerpt from the Wikipedia definition [1]: "A library is also a collection of implementations of behavior, written in terms of a language, that has a well-defined interface by which the behavior is invoked. For instance, people who want to write a higher level program can use a library to make system calls instead of implementing those system calls over and over again. In addition, the behavior is provided for reuse by multiple independent programs."

  The user of your library are other (Eiffel) programmers, that want to generate documents/reports by creating objects from the classes you provide in your library and invoking a "generate" method to create the HTML output from these objects.

  There is no classical user interface, neither graphical, nor in the command line. Since the user of your product is a programmer, the user interface will be an API, which you will design as the second phase of this project.

[1] https://en.wikipedia.org/wiki/Library_(computing)

(Added October 2nd)

- *Question*: Does everything mentioned in the requirements specification have to be implemented?
  Answer: Yes, but I understand that the project description is (to some extent intentionally) ambiguous. The answers to the following questions should clarify most doubts. For any other doubts that you have, please make reasonable assumptions and document them in your requirements specification (as you would in real life).

- *Question*: Can the groups receive a message with all group members of their group?
  Answer: Groups that received an additional group member have been notified via email. All other groups are the same as initially proposed by the groups themselves.

- *Question*: Input file for the project? No parsing of input required, is this correct?
  Answer: There is no import functionality, thus no parsing.
  This might be ambiguous to one specific passage in the project description text: *"It should also be possible to include existing HTML snippets in a document, for instance, an image map already stored in a file or in a string object".*
  For this particular string that is passed from outside there is also no parsing. Its contents should be inserted verbatim in the generated text.

- *Question*: What is the output?
  Answer: Given a document defined in the abstractions given by your library, you need to generate a valid HTML document (for the definition of validity, please refer to the W3C: https://validator.w3.org/)

- *Question*: When will the Reference file of HTML be available?
  Answer:

  Providing an exact HTML document from which you can then extract specifications would probably be too simple for the specifications phase and would not mimic . In general, your library should be able to handle the generation of documents and reports from within other Eiffel programs. Here is a great example of a report being generated from within another program:
  http://pmd.sourceforge.net/pmd-5.2.3/pmd-plsql/surefire-report.html

  Surefire is a plugin that generates reports from test cases (in this particular case, from the PMD project). Ignore the menu on the left and also don't look at the source code just yet, but look at what elements this report required:
  - Headings in different levels (e.g., "Package List", "net.sourceforge.pmd", ...)
  - Tables to report different values (including images to signify status)
  - (Internal) Links + Anchors

  Other elements that are not in this particular example, but are probably also needed to generate documents/reports:

    - Paragraphs
    - (Un-)ordered Lists
    - Text semantics (i.e., bold, italic, underlining)
    - Links to other internal document, links to external URLs

  What you specifically do not to model are (1) styling information (CSS) and (2) JavaScript.

  Please write your specification as general as possible. Your library should be designed in a way that you could easily extend to more HTML elements, or any other document output language. Try to map higher level document elements to a reasonable HTML tag with necessary attributes. However, for some tags it might be necessary to provide certain other attributes. You could model this requirement by giving the users of your library a possibility to add information to document elements in the form of key-value pairs. In HTML this would lead to generating attributes for this element and in Markdown it would not be generated.

  Focusing too much on HTML in your specification and design phase will probably lead to overfitting and will hurt generalizability and maintainability of your model.

- *Question*: Does the user have to know the HTML tags he wants to use?
  Answer: No, see answer in the question before: Build abstractions in a way that a user doesn't even have to know what HTML is. What people will be specifying are higher-level document/report constructs (i.e., heading level 1, table, column with number, etc.) from which your library will then map and generate HTML.

- *Question*: Should the library provide the possibility to include JavaScript and CSS in the HTML output?
  Answer: Good question! No, only HTML tags should be generated.

- *Question*: Is there a limited set of elements to be created by the library (e.g., link, table, page, picture, …) or should it be able to handle every official HTML5 tag?
  Answer: Only a limited subset of HTML tags that can convey document information (e.g., headings, paragraphs, unordered lists, ordered lists, and so on, see the examples above).

- *Question*: Is URL generation and linking only possible between multiple generated HTML pages or should it be possible to link to external pages (e.g., google.com) as well?
  Answer: There should be an abstraction to specify links to external pages as well.

- *Question*: Can we switch the exercise group from Tuesday to Wednesday?
  Answer: If you can find a group that wants to switch with you, both groups should send an email with group number to Jürgen Cito (cito@ifi.uzh.ch) and you can switch.