ACP submission checks

Background

As we had several misunderstandings and easy to avoid mistakes in previous courses this document clarifies most of which occurred, so we have a head start for good submissions.

What must be submitted

A valid submission must contain:

- **Source code**

  In CW2 the sources are going to be marked, so you must include them. If you don't include source code, we are unable to mark

  - **acp_submission_image.tar** as the exported docker image-file inside your ZIP-file on the root level.

The filename acp_submission_image.tar must be exactly that. Invalid filenames will imply a penalty of 3 points, invalid images (which e.g. cannot be loaded or need re-building) imply 8 points.

No image and no option to rebuild or no source code imply 0 points.

Please make sure you are using github for your commit history (and commit in regular intervals), so that in case of need you can proof that your submission has been done before a particular date.

What to check

Every year we have students whose submissions are not being considered because the browser had a problem, something didn't work, etc.

This is very painful, **so please make sure that your submission has been transmitted, and you receive confirmation of this (in Learn by visual means).**

In addition to the physical factor of uploading the submission please check:

- That your submission contains source code and the image-file in tar format

- That your tar file can be loaded into docker
  `docker image load -i acp_submission_image.tar`

- That after you loaded the tar file you can run it
  `docker run -d --publish 8080:8080 --name sXXXX imageId -e ENVIRONMENT-Variables`

  sXXXXX is your student id and the imageId is the result of the image loading – can be either the image tag or the image id (a rather long unique id)

- Check in docker desktop that your image really runs (no exceptions, etc.)
- Check that you can reach the endpoints using postman, curl or any other tool

Especially the last 2 steps are important as otherwise you wouldn't catch more subtle problems like Java version clashes, Spring misconfigurations, etc.

How to create the image-file

You can create a docker image in several ways, yet one which works is like:

- Develop and test your project
- Build the jar within intelliJ by using the maven package command (it will place a jar in the target subdirectory) which you are then picking up again in the docker build process

- On a command line in your solution's directory (where your Dockerfile is located) type
  `docker buildx build --platform=linux/amd64,linux/arm64 -t acp-image-cw1 --load .`

  - This will build an image and load it into docker
  - The image will contain AMD64 (Intel) and ARM64 (Mac) images
  - You will tag it with acp-image-cw1

- You can check your images with `docker images`, and your image should show up top on the list

- Now you can save your image to disk using
  `docker image save acp-image-cw1 -o acp_submission_image.tar`