Project Workflow

Project outline

- In the first week, development teams are defined and projects are assigned.
- Each team is composed of 3 or 4 students.
- The projects end on February 21st, 2020.
- Each team meets the instructor, who acts as *the client*.
- After the first meeting, each team creates the product backlog.
- The instructor reviews and gives feedback on each product backlog.
- After the product backlog is validated, the dev team creates a wireframe and the instructor can be requested to give feedback.
- After the wireframe is validated, the dev team creates an initial model of the database (a database diagram) and the instructor can be requested to give feedback.

The instructor as the client

- During the first meeting, the students ask the client in order to get a picture of the project goals.
- The client participates in a weekly demonstration of the project and gives feedback on it.

Technologies to be used (obligatory)

- Spring Boot: Web, Thymeleaf
- MySQL (e.g. using JDBC or JPA)
- Bootstrap
- Apache Maven
- git + GitHub
- Trello as Kanban board

Technology stretch goals (optional)

- MySQL Workbench for ERD
- Java Persistence API (JPA)
- JUnit
- Spring Security
- Angular / React
- REST
- Docker
- CI/CD

SCRUM

- A sprint is one week long.
- Each team member is a developer. In addition to that, each sprint, the following roles are distributed among the team:
    - Product owner (updates the product backlog)
    - SCRUM master (plans/organizes the meetings)
    - Code reviewer (reviews pull requests)
- Each sprint begins with a sprint planning (planning poker).
- The Kanban board (Backlog, To-Do, In Progress, Blocked, Done) is set up after the sprint planning.
- Every task is described concisely on a card. Implementation should not exceed one day.
- The team members continuously work on the task cards that they pick.
- The daily stand-up can be used as an opportunity to distribute tasks.
- There is a sprint review and retrospective at the end of the sprint (end of the week). Then the burndown-chart is analyzed and discussed.

Git workflow

- The **master** branch is protected, **no commits are pushed to this branch directly!**

- The **dev** branch is protected, **no commits are pushed to this branch directly!**
- Each feature has its own branch, branched from **dev**.
- When a feature is done, the **dev** branch must again be pulled into the feature branch and conflicts must be fixed. This ensures that the feature branch can be pulled into the **dev** branch without conflicts.
- Each sprint, after all features have been merged into **dev** and tested, a pull request from **dev** to **master** is created.
- The instructor accepts and merges the pull request from **dev** to **master** without review.
- A release tag is created on the **master** branch after the pull request has been accepted. Additionally, the release artifacts (current version of the project) are built (e.g. ZIP file, docker images, this will be clarified).

Code reviewing

- As described above, with every sprint start, a code reviewer is assigned.
- The code reviewer must review the pull requests, and either accepts them or ask for changes to be made.
- When changes are requested, the developer of the feature branch is responsible for implementing these.