TI2806: Context Project

Tools for Software Engineering

# Product planning

**Octopeer Viewer**

Group: TSE1

*Borek Beker*
*Marco Boom*
*Leendert van Doorn*
*Ahmet Gudek*
*Daan van der Valk*

April 28, 2016

# 1 Introduction

The Octopeer project focuses on creating an analysis tool for peer reviews on revision management systems such as Github and Bitbucket. The goal of the project is to track a users actions throughout the peer reviewing process and eventually present this information to the user through visualisations, in an easily understandable and highly informative matter. With this data, the users will be able to gain insights on their peer reviewing routines an will have the ability to improve these through time. This document contains the description, requirements and other necessary information required for the creation of Octopeer Viewer, a client-side software responsible for the visualisation of the acquired peer review data.

# 2 Product

The program we are creating will have to adhere to certain requirements and desired features. In this section, a high-level product backlog is given. This backlog lists features the software will or might have at the end of the project. Also a roadmap is created in which our release goals and schedule are depicted.

## 2.1 High-level product backlog

- The product will need an accessible page.
- The page needs to connect to a back-end with the user's data.
- The data will need to be requested from this back-end.
- The received data will need to be mined for useful data.
- The useful data needs to be visualised for the user.
- The visualisations should to be divided into multiple categories.
- The user should be able to only view the desired data category.

## 2.2 Roadmap

At the end of this ten week project we desire to have fully implemented the items listed in section 2.1.

# 3 Product backlog

This sections contains a description of the software's (current) features and defects, technical improvements and know-how acquisition through the use of user stories (If applicable). Additionally, an initial release plan is included.

### 3.1 User stories of features

As a user I want :

- A portal to see data on peer reviews.
- This portal to acquire and visualise my peer review data.
- Insights on the peer reviews I have completed, such as the ...
  - number of peer reviews.
  - length and review times.
- Information about my behaviour while peer reviewing, such as the ...
  - mouse behaviour and clicked components.
  - actual amount of time spent on peer reviewing.
  - external resources used.

### 3.2 User stories of defects

(if applicable)

### 3.3 User stories of technical improvements

(if applicable)

### 3.4 User stories of know-how acquisition

-

### 3.5 Initial release plan

The project is divided into eight weekly sprints with sub-deliverables that we will call milestones. We expect the following functionalities to be implemented on these milestones :
Milestone 1

- A front page is created that can be accessed by the user.
- A connection is established to the database with peer review data.
- GET requests are implemented for data acquisition from the database.
- Demo visuals are created with dummy data which will be replaced with actual user data in the following sprints.

Milestone 2

- A backbone architecture is created.

- Mimicked data is inserted into the database and in turn used for visualisation creation.
- Simple visualisations completed.

# 4 Definition of Done

When a certain aspect of the project is completed by the project team, it will be placed under the "Done" category. The following rules apply for the tasks in this category :

- When a software feature is done, the feature has been coded and tested with unit tests and/or UI tests to a degree deemed acceptable by all team members.
- When a software feature is done, the code has been commented and well documented.
- When a software feature is done, it has been reviewed and accepted by at least two other team members and merged with the main code by a third.
- When a software feature is done and thus has been merged with the main code, the new version of the code has passed all new and previous unit tests.
- A sprint is considered done when all tasks are done according to the points above, or for all uncompleted tasks a valid technical reason is given for not completing it.
- When a sprint is done, all identified bugs on the products of the previous sprints have been fixed.
- The final product is considered done when the main features are implemented and no technically viable requirement remains uncompleted.

# 5 Glossary

-