Software Requirements Specification

Team: StackOverflow Pioneers
Members:
Ungureanu Ionut-Vladimir (TL)
Zoltan Razvan-Daniel
Zeic Alessia-Damaris
Umanschii Ianec
Tocaciu Valeria
Chitimus Mara
Corlaci Andreea

Table of Contents:

1) Introduction

a) **Purpose:**The purpose is building a Conference Management System (CMS) which will help scientists to organize conferences and which will develop a system to make it easier to review papers

b) **Convetions:** We are going to use the following conventions in our project:

- i) CMS - Conference Management System
- ii) ORM - object relation mapping
- iii) DB - database
- iv) SRS - System requirements specification
- v) UC Diagram - The use case UML diagram
- vi) UX - User experience
- vii) UI - User Interface

c) **Intended Audience**: the aim of this project is mostly didactical. However, this would be useful for researchers and scientists including but not limiting to the Computer science domain

d) **Scope:** the target audience will be aided in organizing their work and meetings in a much more “modern” and approachable way

e) **References:**

2) Overall description:

a) **Product perspective**: these are the following informations which will be held

- i) Account details: for each member participating in the conference we will have an account with various user data including the name, positions, etc
- ii) File details: for each submitter we are going to have the ability to store files representing their thesis or the abstract in .pdf or .doc/docx format.
- iii) Bidding process: we are going to have the perspective of users bidding and “winning” proposals.
- iv) Section information: since the CMS is going to have a conference activity section, mainly papers presentation, we will need to store various informations regarding each section

b) **Product features for each user class and characteristics:** Users of the system should be able to participate and/or manage conferences using this CMS. An user should be able to:

- i) Create an account of various types:
  - (1) Author
  - (2) Reviewer
  - (3) Listener
- ii) Login into the system
- iii) Update their profile
- iv) Join conferences or Create them

v) Upload their paper and abstract

vi) Only for authors:

(1) Submit proposals

(2) Make presentations

vii) Only reviewers:

(1) Bid proposals

(2) Review papers

viii) Only for chairs:

(1) “Tie break” equally matched proposals

c) Operating Environment:

i) Database

ii) Backend made of Java

iii) Front end made of Javascript

iv) Client/server system for communication between front and back end

v) Will work on any browser from UX point of view

d) Design and Implementation constraints:

i) SQL commands for the database

ii) REST API for the back end which will communicate with the front end

iii) Database implementation using the given tools

iv) Front end implementation using the mentioned framework

e) Assumptions and dependencies:

We will assume the following things:

- the user is familiar with and uses the latest version of Google Chrome, Mozilla Firefox, Opera or any browser of their choice supporting the latest version of JavaScript;
- the user will submit correct data both in the account registration phase and the paper registration phase;

3) Functional requirements:

The following steps represent the lifecycle of a conference according to our CMS application:

a) A conference participant can register on the CMS. The registration steps include paying the tax and validation of the data.

b) An author can submit the proposal (the abstract and the paper), which will be validated.

c) The bidding process can start. Each reviewer can bid on the proposal they want to examine.

d) The Chair and the Co-Chair can choose the reviewers for each paper.

e) Every paper can be reviewed. After the process, each paper can be either accepted or rejected. In case of a tie break, the Chair can decide if the paper is accepted or not

f) Each section can be supervised either by a Chair or Co-Chair.

g) The speaker can present its paper in front of the participants (listeners), he can also upload a presentation.

4) Nonfunctional requirements:

a) Use case diagram (the one you guys “reviewed” will be put here)

b) Database normalization: all tables must be in 3NF in order to reduce data redundancy and in order to optimize the storage of the data. However, the most important reason for imposing such a condition is to avoid CRUD anomalies which will interfere heavily with our transactions.

c) Respecting the SOLID programming principles:

   i) Single responsibility principle

   ii) Open/Closed principle

   iii) Liskov Substitution Principle

   iv) Interface Segregation Principle

   v) Dependency Inversion Principle