# The Voting Plugin for Redmine

## Requirements Contract

Tural Mammadov

Maxat Nukhayev

Oleksii Voitenko

Gellert Kispal

Furkat Kochkarov

Yuliia Brendel

Alla Kornoukhova

Tutor: **Alex Schlosser**

Customer: **Vitalii Avdiienko, Konstantin Kuznetsov**

## Project description

The Software engineering course at Saarland University is administered yearly with over 300 participants. During this course, every student is expected to work in groups in one specific project. The projects are defined by different clients from across the world. Due to the high number of participants, matching students to desired projects becomes a complicated problem. Another layer of complexity is assigning tutors to mentor projects, since tutors also have the ability to specify their preferred projects as well.

Our solution is designed to simplify the process of assigning students and tutors to projects for the administration staff of the Software Engineering course. Student and tutors will be provided an interface to vote for a number of desired projects. This list of preferences is used during the assignment process. Students and tutors will be assigned to projects automatically based on a matching algorithm.

The current solution to this problem is incomplete and requires lots of manual input. All topics have to be entered into the system manually. All students registered are manually copied from the system in order to provide input to an optimization algorithm. These result are then manually verified and re-entered into the Redmine system. Then all project have to be manually created with a list of students, tutors and customers. An automation of the above described procedure will save the administrators time and effort and ensure less tedious work.

## Glossary

1. **User:** Any person that is registered with the Redmine system is considered a <User>.

2. **Privilege** - The <User's> access rights within Redmine system. Here is a list of privileges:
    - **Admin-**role of the <User> as a manager.
    - **Tutor-**role of the <User> with ability of voting for the <Topic> as a mentor.
    - **Student-**role of the <User> with ability of voting for the <Topic> as a participant.

3. **Voting poll -** A set of topics that the <User> may specify a preference to.

4. **Selection** - Set of <User> preferences sorted by configurable pattern.

5. **Voting procedure** - Action of signing up the <User's> selection.

6. **Voting pattern** - The parameters of which Student can vote for a project. For example one could define a pattern that forces every Student to for vote a minimum number of desirable project and undesirable projects.

7. **Shuffle resolver** - Is an algorithm that map's <User> to <Topic> based on their <Selection>.

8. **Project** - group of people working on the same topic. All projects will have a list of assigned <Students> and <Tutor>.

9. **Course -** Entity that defines configurable parameters such as registration deadline,voting deadline,maximum project participants, minimum project participants and contains set of related <Users>,<Voting poll> and <Projects>.

## MUST-HAVE Features

Functional must-have features:

1. **Extended user accounting system.** Voting plugin architecture is built on top of the existing user accounting system implemented in Redmine. Our approach will define at least three types of access privileges: Admin, Tutor, Student. User's role defines his capabilities within a system. Any User registered within Redmine system receives Student role by default.

2. **Voting procedure**. Voting start-end deadline dates are set up by Admin. Students has ability to vote as a group. Student and Tutor has the ability to rate topics. Student's selection can be changed before the deadline. Topic distribution will be organized only over the students who participated in the voting. If the user missed the deadline he will not be assigned to any projects. List of such student is reported to the Admin.

3. **Automatic User-to-Project assignment**. User-to-Project assignment will be done by automatic Shuffle resolver, considering that no User may be assigned to the same project both as Student and Tutor. Shuffle resolver is launched manually by Admin. Admin observes and can manipulate all assignments produced by Shuffle resolver and submits them afterwards.

4. **Automatic project creation**. After Admin approves Student-to-Project assignment results corresponding Redmine Projects will be created. Students will be automatically assigned to their Projects.

5. **Manual setup.** Admin has to manually add topics to the voting poll, set up voting procedure parameters (registration deadline, maximum number of members per project, voting deadline).

6. **Project management**. Admin can perform following actions on Projects:
   - Create Project
   - Delete Project
   - Assign Topic to Project
   - Assign Student to Project if number of current Project participants is less than maximum Project members.
   - Remove Student from Project.
   - Swap Students between Projects.

- Assign Tutor to Project if there is no Tutor already assigned.
- Remove Tutor from Project.
- Swap Tutors Between Projects.

7. **Manual editing**. Admin can manually remove or edit topics from the voting poll. Admin can manually register/unregister Users anytime. Admin can manually assign new role to any User. Admin can change course parameters (registration deadline, maximum number of members per project, voting deadline).

8. **Short project description**. During voting procedure there will be short descriptions available for Student.

Nonfunctional must-have features:

9. Plugin will be written on the Ruby on Rails framework (Ruby 2.1.5p273 x86_64 | Rails 4.2.4 ). Plugin will be designed to operate in Redmine Ver 3.1.2 and Database management system MySQL 5.5.46

10. Plugin page layout will be set up for the Google Chrome, Mozilla Firefox desktop edition browsers.

11. User interface will support English language.

12. In order to communicate with plugin user needs connection with university network..

13. Online documentation will be supplied.

## MAY-HAVE Features

**Grouping Features: This set of features allows students to be grouped:**

14. **Students can create and and join groups.** Every student has the right to create a group, however not all students need to create one. Therefore one will also have the functionality to delete their group or unregister from a group. Most students will be members of a group created by someone else. Thus only the student that created the group can vote on a specific topic.

15. **Group creator can accept group join request or remove users from the group before the deadline.** In order to organize all students to in a meaningful way the student that originally created the group can accept new member or remove someone from the group.

16. **Group leader can delegate his status to a group member.** In the case where a group leader decides they want to join a different project, they can delegate their function to a different student that is also a member of their group.

17. **Voting for team leader after project group establishment.** Once student are in a project they are able to elect their project manager.

**Ease of Use Features: This set of features deals with the usability of the system:**

18. **User Interface with Drag and Drop functionality.** Making the voting procedure appealing and user friendly a drag and drop method can be used. Instead of manually assigning number to project the users will be able to drag and drop specific projects into boxes labeled "desired" and "undesired".

19. **Grace period after the deadline for already logged in users.** A period of about 15-20 minutes will be granted to users who are already signed into the system and are performing the voting procedure. This will ensure that a more complete voting and caters to students who like to complete their duties at the very last minute. (Which is not very uncommon in computer science).

20. **Balance teams in Shuffle resolver according to their skills and background.** Users can specify their strengths and skills which the algorithm will take into consideration when running the resolver. This feature will assure that students with the desired skill will be assigned to a project that better suits their abilities.

21. **Topic will be supplied with detailed description.** All topics will have full description of the project.

**Statistical Analysis Features: Set of features dealing with statistical measurements:**

22. **General statistics generation by the end of the term.** These statistics would be able to indicate which projects were successfully completed during the course and which projects were deleted because lack of student engagement. Grades which each individual project has received can also be tracked

23. **Real-time voting statistics displayed to admin.** During the voting period the admin can see in real time which are the most popular topics. Popularity is based on the number of students that voted on the topic and the weight of which they voted for them.
24. **General statistics generation by the end of the term.** These statistics would be able to indicate which projects were successfully completed during the course and which

projects were deleted because lack of student engagement. Grades which each individual project has received can also be tracked.

25. **Export** statistics as a pdf file

**Administrative Features: Set of features concerning the administration of the course:**

26. **Additional voting rounds initiated by admin.** In case some students join the course late, the admin can initiate an additional voting round. This voting round will not consider group voting, each student has to vote individually. The additional voting round will display remaining available topics. These projects are that ones that have not met the maximum number of participants.

**27.** **Populate voting poll from file.** Admin has the ability to list all topics in an file and upload to the system. This procedure will automatically create projects in the system based on the contents of the file.

28. **Add customer to the system.** The system considers an additional role: Customer. Customer has ability to suggest tasks. All tasks will be added to the voting poll after admin confirmation. This user will have the ability to see the list of students working on their created project.

29. **Notification system will use mailing list to send personal or broadcast messages.** In order to keep tabs on student groups an admin can easily message each group from the system or message a set of desired individuals. Admin has access to notification system.

30. Tutor’s contract workload affects the number of projects he would be assigned to.

31. **Searching by User on arbitrary attribute.** In order to manage students quickly and efficiently one could search for any student using different attributes such as matriculation number, last name, email etc.

32. **Student that had done the project before could skip voting.** In case Student is retaking the course or just wants to participate in writing the exam Student is able to specify this in the system, skip voting and also not take part in any project. In this case User accounting System is extended with a new role: Senior Student.

33. **Admin can lock / unlock User account.** In case some students drop the course, an administrator can indicate this by having their account locked.

34. **Student experience is taken into account during assignment to a topic.** Prior to the voting Student has to participate in private survey where Student points to skills that will be considered during assignment.

**Nonfunctional may have features**

35. Plugin will operate in Microsoft Edge desktop edition browser.

36. Plugin will operate in Safari desktop edition browser.

37. Database backup available to administrator

## MUST-NOT HAVE Features

38. User interface supports additional languages in current version. Multiple localizations are not considered in current release.

39. Plugin operation supports any versions of Redmine.
Our system will only run on the specified version of Redmine in the MUST-Have section. Support for other versions can be extended in later releases.

40. Plugin operation supports any versions of Ruby language and Rails framework.
We should remove this since the customer should not care about what language we implement our solution in. Support for other versions can be extended in later releases.

41. Plugin operation considers maintainability by developers. We cannot guarantee that our system will include features not specified in the contract. Maintenance of this system is outside of scope of this project.

42. User has ability to run SQL command from plugin environment.
This feature can be implemented in later releases.

43. User can un-register himself/herself from the system.

All features not explicitly mentioned in current contract are not guaranteed. Plugin is licensed under Proprietary License. Assignee will receive rights to use, change the source code, but is not allowed to sell, distribute or license to third parties.