DOSEMS - Project report

Introduction

Our project, DOSE Management System, have been a fun but sometimes challenging project. Besides all the normal barriers that arise due to the fact that this is a distributed software project, new technologies on both ends played their part.
On the front end we choose to develop using the AngularJS framework. This was new to all in the front end team, although some prior knowledge of web development existed. A common feeling was that the more we learned, the more we wanted to redo everything we had already done, but due to time constraints this was not always possible.
On the back end team development using Eiffel was required. They had some prior knowledge thanks to a small local project which aided in the learning and development of the back end.

Progress

Functional requirements

1. Product backlog
    a. Status : Done
    b. Comments : In this view, one can find the sprints and tasks for that project and also the users of that project. Tasks can be created, edited or deleted and this view links to the manage project and manage users view.
2. Sprint backlog
    a. Status : In progress
    b. Comments : We assign the sprint automatically when the task is created.
3. User registration
    a. Status : Done
    b. Comments : From the login page, a new user can sign up for the system. After registration the user is returned to the login screen to make sure it can actually login.
4. Appointing users to project
    a. Status : In progress
    b. Comments : Right the system can return the current role of user in a project, but the role can't be set. We don't send a notification to the appointed user.
5. Basic user management
    a. Status : Done
    b. Comments : We made the choice to have username == email, so we have one field less in the edit user view.
6. Support for different roles
    a. Status : In progress

    b. Comments : As mentioned in Req 4, we can get the role of a user, but can’t set it at this moment.

7. User authentication
    a. Status : Done
    b. Comments : The email needed for signup mu be unique in the system, and the user must signup with a password at least 3 characters long, that is stored as a hash in the DB for extra security.
8. Assign users to requirements
    a. Status : In progress
    b. Comments : A user can be assigned to a task, but now anyone can assign a user, and anyone can change status of the task.
9. Requirements decomposition
    a. Status : Not implemented
    b. Comments :
10. Labor input counting
    a. Status : In progress
    b. Comments : We can assign points to the tasks and requirements, but not set them to done.
11. Developers chart
    a. Status : Done
    b. Comments : From the project dashboard it is possible to get a ranked list ot the developers in the project
12. Users dashboard
    a. Status : In progress
    b. Comments : Right now lists the current projects, but not the deadlines for the tasks
13. Statistical reports
    a. Status : Not implemented
14. Team communication tools
    a. Status : Not implemented
15. Integration with issue trackers
    a. Status : Not implemented

Non-Functional requirements

16. Maintainability and Extensibility
    a. Status : In progress
    b. Comments : Since we are working with new techniques we learn more and more what are the best practices and how to structure code. Some refactoring have been done, but not as much as would be needed, at least for the front end team. The backend have comments to describe the flow of the code.
17. Security
    a. Status : In progress
    b. Comments : The roles exists, but they need to be used more extensively in order to get the tight level of security.

18. Robustness
    a. Status : In progress
    b. Comments : In as many places as possible, the system gives feedback to the user when input is incorrect, or when an unknown page is requested. Server errors aren’t handled at the moment.

Project reflection

Front end team

As part of the front end team I can say that this has been an interesting project where I have learned a lot, not only about the struggles and rewards of a distributed development, but also team coordination, work structure and a new framework, AngularJS.
Regarding the fact that this is our first project of this type, everything is new, and it was hard to know how to plan the workload division. The fact that time differences played a part in communication was something that we perhaps was not good at taking into account in the beginning, but after a while we found a good rhythm.
I think development have worked well, since culture differences aren't that great, and that both teams are part of the developers culture as well.
Our biggest lesson from this would be to at least keep something familiar when starting a big new project to help get a good structure and aid planning.
We as a team were all excited to try out AngularJS as a framework, but it might have been wiser to choose something we knew from the start. We realised that implementation took longer than estimated in many cases, and that our primary structure of the site and division of work could have been improved.
Nevertheless, this has been a very fun experience, a great way to test work in a distributed environment, meeting new people and cultures, and see that is actually is possible to work together from across the globe!

Back end team

This was our first time working on a project of this magnitude, and it resulted very exciting and interesting. From one side, it was an excellent opportunity to get in contact with people from abroad, a real challenge for the organization and coordination of the activities to carry out this project, without mention meeting great and interesting people. This experience was very useful as a little example of a real life software development system: we had to learn to communicate with our partners, discuss the best alternatives for the different parts of the project, make important decisions involving the structure, design and implementation of the system, and, why not?, improve our english skills.
Regarding to the implementation part, this was also our first time with Eiffel, and it was a bit challenging at the beginning, as always happens with every new language one learns, but we

could get along with it and take advantage of part of its potential (all that we could discover). It supported us very comfortable ways and tools that allowed us to organize our code to make it the most readable and understandable as possible, such as the feature organization via comments and its amazing contract management. Personally, I can say that Eiffel can be very comfortable to work with when you get to know it.

The distance between us maybe made our work much complicated as expected, but it also made us learn to work in a different way, with different people, different cultures, and that was very positive, since it taught us new ways to open our minds. This was a fabulous experience that is not possible to have everyday, so we wanna thank DOSE and our excellent team partners (and friends) for this spectacular adventure.